Planning Text for Advisory Dialogues: Capturing Intentional and Rhetorical Information

Planning Text for Advisory Dialogues: Capturing Intentional and Rhetorical Information

Johanna D. Moore*
University of Pittsburgh

Cécile L. Paris†
USC/Information Sciences Institute

To participate in a dialogue a system must be capable of reasoning about its own previous utterances. Follow-up questions must be interpreted in the context of the ongoing conversation, and the system's previous contributions form part of this context. Furthermore, if a system is to be able to clarify misunderstood explanations or to elaborate on prior explanations, it must understand what it has conveyed in prior explanations. Previous approaches to generating multisentential texts have relied solely on rhetorical structuring techniques. In this paper, we argue that, to handle explanation dialogues successfully, a discourse model must include information about the intended effect of individual parts of the text on the hearer, as well as how the parts relate to one another rhetorically. We present a text planner that records this information and show how the resulting structure is used to respond appropriately to a follow-up question.

1. Introduction

Explanation systems must produce multisentential texts, including justifications of their actions, descriptions of their problem-solving strategies, and definitions of the terms they use. Previous research in natural language generation has shown that schemata of rhetorical predicates (McKeown 1985; McCoy 1989; Paris 1988) or rhetorical relations (Hovy 1991) can be used to capture the structure of coherent multisentential texts. Schemata are scriptlike entities that encode standard patterns of discourse structure. Associating a schema with a communicative goal allows a system to generate a text that achieves the goal. However, we have found that schemata are insufficient as a discourse model for advisory dialogues. Although they encode standard patterns of discourse structure, schemata do not include a representation of the intended effects of the components of a schema, nor how these intentions are related to one another or to the rhetorical structure of the text. While this may not present a problem for systems that generate one-shot explanations, it is a serious limitation in a system intended to participate in a dialogue where users can, and frequently do, ask follow-up questions.

In this paper, we argue that to participate in explanation dialogues successfully, a generation system must represent and reason about the intended effect of individual parts of the text on the hearer, as well as how the parts relate to one another rhetorically. We present a text planner that constructs explanations based on the intentions of the speaker at each step and that notes the rhetorical relation that holds between each pair of text spans. By recording the planning process behind the system's utterances as well as the user's utterances in a dialogue history, our system is able to reason about its

* Department of Computer Science and Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA 15260. E-mail: jmoore@cs.pitt.edu
† USC/Information Sciences Institute, 4676 Admiralty Way, Marina del Rey, CA 90292-6695. E-mail: paris@isi.edu

previous utterances both to interpret and to answer users' follow-up questions. We describe the plan language employed and the plan structure built by our system and provide an example of how this structure is used in responding appropriately to a follow-up question. Additional examples appear in Moore and Swartout (1989) and Moore (in press).

2. Motivation: A Naturally Occurring Advisory Dialogue

When we began our work on interactive explanations, we gathered samples of naturally occurring dialogues from several sources: transcripts of electronic dialogues between system users and operators collected by Robinson (1984), protocols of programmers interacting with a mock program enhancement advisor, and tape recordings of office-hour interactions between first-year computer science students and teaching assistants. A portion of a dialogue extracted from the office-hour interactions appears in Figure 1.

In this dialogue, a student and a teaching assistant are discussing a programming assignment that involves writing a procedure to swap the values stored at two locations in the C programming language. The student is confused about how to write the procedure because he does not understand that C is a call-by-value language, and so he must pass the addresses of the two variables to be swapped. In the teacher's response in turn 8, she explains that in C one cannot change the value of a variable defined outside of a procedure. She justifies this by saying that C is call-by-value and then goes on to define this term. The student then asks "What's call-by-value?" (turn 9). To respond appropriately to this question, the teacher must realize that she has defined call-by-value in abstract terms as part of her previous explanation and that her first attempt was not fully understood. In this dialogue, the instructor recovers from this failure by giving a very specific example of how call-by-value works (turn 10). The teacher explains call-by-value differently the second time because she realizes that she has tried to explain this once before, and that the strategy she used the first time was not sufficient.

Note that giving a very general description of a concept is a good first strategy. General definitions are an efficient way to convey knowledge because if the hearer understands a general definition, it covers a wide range of cases. Giving example(s) first would be a bad approach because this strategy requires the hearer to form the correct generalization from the example(s). Unless the examples are numerous enough and well chosen, this may be difficult to achieve (Klausmeier 1976). However, when a general description is not understood by itself, examples provide effective elaboration and may help the hearer assimilate the general description (Charney, Reder, and Wells 1988). In the sample dialogue, because the teacher knows that she has already given a general description in turn 8, she can give a very specific example in turn 10.

Another characteristic of advisory dialogues is exemplified in this sample. Advice-seekers often ask questions such as "Why?" and "How come?". To interpret such questions, the system must have a representation of what it has said just prior to the query, so that it can determine what the user could be asking about.

3. Limitations of Previous Approaches

There are two main approaches to text generation. In attempting to construct a system capable of participating in dialogues like the one above, we found that neither approach could be directly applied to this task. Here we describe the limitations of these approaches for our purposes.

STUDENT	I can't figure out how to write the SWAP procedure in C.	[1]
TEACHER	You write a procedure called SWAP that takes the addresses of the two variables to be swapped as its arguments.	[2]
STUDENT	I did that.	[3]
	⋮	
[Student shows teacher his program code.]		
	⋮	
TEACHER	Here [points to call to SWAP in the main program], when you call SWAP, you have to pass the *addresses* of the variables you want swapped.	[4]
STUDENT	Why?	[5]
TEACHER	Because otherwise the procedure just swaps its local copies of the arguments.	[6]
STUDENT	How come?	[7]
TEACHER	You cannot change the value of a variable defined outside a procedure unless you pass its address. This is because C is call-by-value. That means that when you pass an argument to a procedure in C, the procedure makes a copy of the argument, and does its work on that copy. The value of the variable passed in as an argument remains unchanged.	[8]
STUDENT	What's call-by-value?	[9]
TEACHER	OK, suppose in our main program we have a variable V with value 3, and a procedure called PLUS-ONE that takes one argument, call it A. Now when we call PLUS-ONE(V) what actually happens is that the value of V, which is 3, is copied to A. So A equals 3 and our procedure adds 1 to A, so A now equals 4. The original variable V that appeared in the call to PLUS-ONE does not change. V still equals 3.	[10]
STUDENT	So, how can I ever write a procedure that swaps two values?	[11]
TEACHER	You write a procedure that takes addresses of variables as its arguments. When you call it, you pass the pointers to the variables to be swapped. So you write SWAP(&X,&Y). The & sign means take the address of the variable. When you say &X, the address of X is copied to a temporary variable, and SWAP uses the temporary copy of the address. But the address always points to the same place, namely the location where X is held. So SWAP will operate on X, using a copy of X's address, rather than operating on a copy of X.	[12]
STUDENT	Oh, okay.	[13]

Figure 1
Portion of a naturally occurring advisory dialogue.

3.1 Planning with Speech Acts

Work by Cohen and Perrault (1979), Appelt (1985), and Cohen and Levesque (1990) demonstrated that planning techniques could be useful in text generation. These researchers provide a formal axiomatization of illocutionary actions that may be used to reason about the beliefs of the hearer and speaker and the effects of surface speech acts on these beliefs. To use this approach in a generation process, the system first generates hypotheses about what combinations of actions to perform. For efficiency, Appelt (1985) uses simplified versions of the axioms (called **action summaries**) encoded in NOAH-style plan operators (Sacerdoti 1977) to generate these hypotheses. Theorem-proving is then used to determine if a series of proposed actions will have the desired effect on the hearer's mental state. The systems that have been built within

this framework to date (Cohen 1978; Appelt 1985) plan short (one- or two-sentence) texts to achieve the speakers' goal(s).

In this approach, the **intentional structure** describing the speaker's purposes and the relationships between them (Grosz and Sidner 1986) is explicitly represented. However, this approach does not represent or use rhetorical knowledge about how speech acts may be combined into larger bodies of coherent text to achieve a speaker's goals. It assumes that appropriate axioms could be added to generate longer texts, and that the text produced will be coherent as a byproduct of the planning process. However, this has not been demonstrated.

Moreover, we believe that building a system to produce multisentential texts directly from the logics proposed by proponents of this approach would prove to be computationally infeasible. We see two problems. First, this approach requires the system to acquire and maintain a correct, detailed model of the hearer's beliefs. Sparck Jones (1989) has questioned not only the feasibility of acquiring such a model, but also of verifying its correctness, and the tractability of utilizing such a model to affect a system's reasoning and the generation of responses. Second, all of the formal axiomatizations espoused by proponents of this approach are based on extensions to first-order logic. In the general case, theorem-proving in first-order logic is undecidable. To be fair, some proponents of this approach, e.g. Cohen and Levesque (1990), claim to provide a specification *of* an agent, and do not claim that the axiomatization should be used directly as a specification *for* the implementation of an agent. Heuristics are clearly needed in order to make an implementation based on such a formalism tractable. Our approach employs rhetorical strategies as compiled knowledge about what actions may be used to satisfy certain intentions. In fact, our operators have much in common with Appelt's action summaries.

3.2 The Schema-Based Approach

To produce the longer bodies of text required for advisory dialogues in an efficient manner, other researchers turned to an approach that makes use of script-like structures, **schemata**, to generate coherent multisentential texts achieving a given communicative goal. Schemata, originally proposed by McKeown (1985), represent standard patterns of discourse structure by encoding the set of communicative techniques that a speaker can use for a particular discourse purpose. Schemata are made up of **rhetorical predicates** that characterize the means that speakers use to achieve their goals and delineate the structural relations between propositions in a text. Linguists, e.g., (Shepherd 1926; Grimes 1975), found that rhetorical predicates tend to occur in certain combinations, and McKeown further observed that certain combinations are more appropriate than others depending on the discourse purpose. For example, she found that speakers frequently describe objects by:

1. Identifying the object as a member of some generic class and giving attributive or functional information about the object.
2. Providing analogical, constituent, or additional attributive information about the object.
3. Providing examples of the object.

To encode these standard patterns of discourse structure, McKeown devised several schemata that represent combinations of rhetorical predicates. For example, the above pattern is embodied in the IDENTIFICATION **schema**, shown in Figure 2. By associating each rhetorical predicate with an access function for an underlying knowledge base,

Identification Schema[1]

Identification (class & attributive/function)
{Analogy/Constituency/Attributive/Renaming/Amplification}*
Particular-illustration/Evidence+
{Amplification/Analogy/Attributive}
{Particular-illustration/Evidence}

Sample Definition Generated using Identification Schema:

(1) A ship is a water-going vehicle that travels on the surface. (2) Its surface-going capabilities are provided by the DB attributes DISPLACEMENT and DRAFT. (3) Other DB attributes of the ship include MAXIMUM_SPEED, PROPULSION, FUEL (FUEL_CAPACITY and FUEL_TYPE), DIMENSIONS, SPEED_DEPENDENT_RANGE and OFFICIAL_NAME. (4) The DOWNES, for example, has MAXIMUM_SPEED of 29, PROPULSION of STMTURGRD, FUEL of 810 (FUEL_CAPACITY) and BNKR (FUEL_TYPE), DIMENSIONS of 25 (DRAFT), 46 (BEAM), and 438 (LENGTH) and SPEED_DEPENDENT_RANGE of 4200 (ECONOMIC_RANGE) and 2200 (ENDURANCE_RANGE).

Figure 2
TEXT identification schema and sample generated text (from McKeown [1985], pp. 210–212).

these schemata can be used to guide both the selection of content and its organization into a coherent text. Figure 2 also shows a sample text generated from a knowledge base of naval concepts using the IDENTIFICATION schema. McKeown identified four schemata, each of which could be used to achieve one or more discourse purposes.

As shown in Figure 2, schemata contain many options and alternatives. To instantiate a schema, its components are filled in sequentially by using the access functions to search the underlying knowledge base for information that satisfies the rhetorical predicates. In McKeown's theory, each entry in the schema can be filled by an instantiated predicate or a full schema of the same name. So, for example, in the IDENTIFICATION schema, the first entry can either be satisfied by an instance of the IDENTIFICATION predicate, or a recursive instantiation of the IDENTIFICATION schema itself.

McKeown found that schemata alone were not sufficient to constrain the generation process. To overcome this, when a schema indicates that more than one choice is possible, McKeown's system appeals to constraints on the shift of focus of attention (Sidner 1979). These constraints guide the selection of the information that fits in best with the previous discourse. Since McKeown's seminal work, many other researchers have used schemata as the basis for producing multisentential texts. In many cases, these researchers found that schemata provided only a partial solution, and they have identified additional factors that control the generation process: Paris (1988) uses information about the user's knowledge of domain concepts to tailor descriptions of complex physical objects to a particular user; McCoy (1989) uses object perspectives and a user model to provide corrective responses to users' misconceptions about ob-

1 The "{}" indicate optionality, "/" indicates alternative, "+" indicates that the item may appear one or more times, and "*" indicates that the item may appear zero or more times.

SYSTEM	What characteristics of the program would you like to enhance?	[1]
USER	Readability and maintainability.	[2]
SYSTEM	You should replace (SETQ X 1) with (SETF X 1). SETQ can only be used to assign a value to a simple-variable. In contrast, SETF can be used to assign a value to any generalized-variable. A generalized-variable is a storage location that can be named by any access function.	[3]

Figure 3
Partial dialogue.

jects; and Hovy (1988) uses pragmatic and stylistic information to produce different accounts of the same incident.

3.2.1 Inadequacies of Schemata for Advisory Dialogue. Like others, we found that schemata were not sufficient to handle the issues we wished to investigate. When we attempted to use schemata for our purposes, two main problems arose. First, schemata lack an explicit representation of the intentional structure of the text being produced, and therefore are missing the information needed to recover from explanatory failures. Second, we found that schemata are too rigid to handle certain of the opportunistic phenomena we observed in naturally occurring dialogues. We discuss these two problems in more detail.

Lack of Intentional Structure. As we have seen, schemata encode standard patterns of discourse structure. However, they do not include an explicit representation of the effects that individual components of a schema are intended to have on the hearer, or of how these intentions relate to one another or to the rhetorical structure of the text. This presents a serious problem for a system that must participate in a dialogue where users can ask follow-up questions like the ones we saw in the sample dialogue of Figure 1. If a system does not keep a record of the intentions behind its utterances, it cannot determine what went wrong when the user indicates that an explanation was not completely understood, nor provide an alternative explanation to correct the problem.

To allow a system to handle follow-up questions that may arise if the user does not fully understand an explanation, a generation facility must be able to determine what portion of the text failed to achieve its intended purpose. If the generation system only knows the *top-level* communicative goal that was being achieved by the text (e.g., to make the hearer know a concept, or to make the hearer want to perform an action), and not what effect the individual parts of the text were intended to have on the hearer or how they fit together to achieve this top-level goal, its only recourse is to use a different strategy to achieve the top-level goal. It is not able to re-explain or clarify any *part* of the explanation.

We illustrate this important point by working through an example taken from an actual dialogue with a system called the Program Enhancement Advisor (PEA) (Neches, Swartout, and Moore 1985). (Note that, for precisely the reasons we describe in this paper, PEA does *not* employ schemata to generate its utterances. We describe PEA's text planner in Section 5.) As shown in Figure 3, PEA begins its interaction with the user by asking what characteristics of the user's program are to be enhanced and then suggests changes that will improve these aspects of the program. Now consider what a schema that could produce the system's utterance in turn 3 of the sample dialogue in Figure 3 would look like. One schema that would suffice, which we have called the *Recommend-Replacement Schema*, is shown instantiated in Figure 4.

System's Utterance

(1) You should replace (SETQ X 1) with (SETF X 1). (2) SETQ can only be used to assign a value to a simple-variable. (3) SETF can be used to assign a value to any generalized-variable. (4) A generalized-variable is a storage location that can be named by any access function.

Recommend-Replacement Schema (Instantiated)

(Recommendation (replace-setq-with-setf)) (1)
(Compare&Contrast-Attributive)
(Attributive SETQ use assign-value-to-simple-variable) (2)
(Attributive SETF use assign-value-to-generalized-variable) (3)
(Identification generalized-variable storage-location (4)
(restrictive named-by access-function))

Figure 4
Hypothetical schema representation of system's utterance.

This schema begins with a RECOMMENDATION of a replacement act, followed by a COMPARE & CONTRAST predicate that highlights the important difference(s) between the replacee and the replacer.[2] Instead of using a simple predicate, we instantiate the COMPARE & CONTRAST predicate using a schema that expands into two ATTRIBUTIVE predicates and an IDENTIFICATION predicate that defines a term introduced in the second ATTRIBUTIVE.

Note that this schema indicates *what* to do *when*, i.e., recommend the action and contrast the replacee with replacer, but it does not say *why* this information is being presented. For example, the schema does not indicate that, by contrasting SETQ with SETF, the speaker is trying to persuade the hearer to do the replace act. Nor does it indicate that the text produced by the IDENTIFICATION predicate appears because the speaker is trying to make the hearer know about the concept generalized-variable. In addition, the relationships between these intentions are not represented. To make clear what is missing, we have represented in Figure 5 the intentional structure of this text using Grosz and Sidner's (1986) notions of dominance and satisfaction-precedence. In Grosz and Sidner's theory (1986, p. 179), if an action that satisfies one intention, I_1, is intended to provide part of the satisfaction of another intention, I_2, then I_2 *dominates* I_1. I_1 *satisfaction-precedes* I_2 whenever I_1 must be satisfied before I_2. The representation shown in Figure 5 makes it clear that the expert system's (E) top-level intention (I_0) is to get the user (U) to intend to replace (SETQ X 1) with (SETF X 1), and this intention dominates E's intentions to recommend this act (I_1) and to persuade U to perform it (I_2). In addition, for this schema, the recommendation (I_1) must be satisfied before the persuade (I_2) is attempted.

A schema can be viewed as the result of a "compilation" process where the *rationale* for all of the steps in the process has been compiled out. What remains is the top-level communicative goal that invoked the schema (in this case something like *Get the user to adopt the goal of replacing* SETQ *with* SETF), and the sequence of actions (i.e. instantiated

2 The first step of the schema is to identify commonalities of the two entities being contrasted. In McKeown (1985), this step is optional if no commonalities exist. We have changed this definition slightly to render this step optional if the speaker believes these commonalities are known to the hearer. This is the case here, so the instantiated schema does not contain this step.

System's Utterance:

You should replace `(SETQ X 1)` with `(SETF X 1)`. `SETQ` can only be used to assign a value to a simple-variable. `SETF` can be used to assign a value to any generalized-variable. A generalized-variable is a storage location that can be named by any access function.

Intentional Structure:

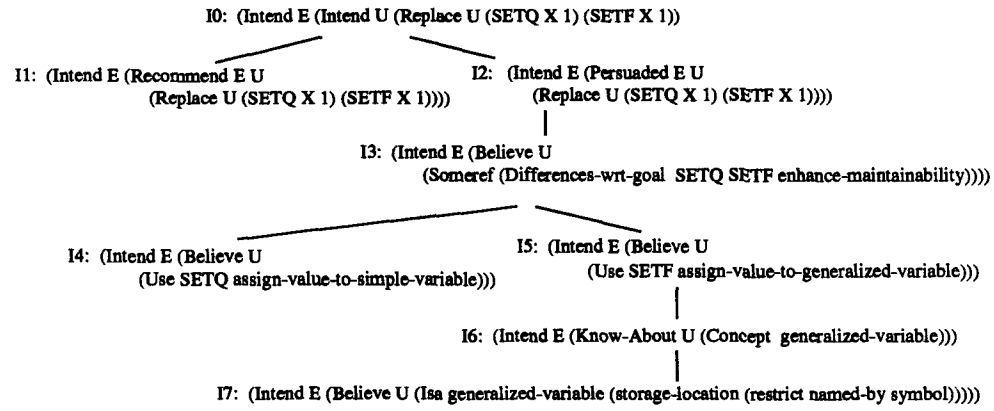

Figure 5
Intentional structure of system's utterance.

rhetorical predicates that cause sentences to be generated) that are used to achieve that goal. All of the intermediate structure shown in Figure 5 has been lost.[3]

Because of this compilation, schemata provide a computationally efficient way to produce multisentential texts for achieving discourse purposes. They are "recipes" of rhetorical actions that encode frequently occurring patterns of discourse structure. Using schemata, the system need not reason directly about how speech acts affect the beliefs of the hearer and speaker, nor about the effects of juxtaposing speech acts. The system is guaranteed that each schema will lead to a coherent text that achieves the specified discourse purpose.

However, this compilation is also a disadvantage. If the hearer does not understand the utterance produced by a schema, it is very difficult for the system to recover. In the example above, without understanding why the COMPARE & CONTRAST schema and IDENTIFICATION predicate are present in the instantiated structure, the system cannot determine how to proceed if the user does not immediately understand and accept the system's utterance. In the sample text under consideration, the COMPARE & CONTRAST occurs as part of the top-level schema in order to persuade the hearer to perform the recommended replacement action. This is represented in the intentional representation of Figure 5 by intentions I_2 and I_3 and the dominance relationship $dominates(I_2, I_3)$.

3 Note that the schema shown in Figure 4 would not be compiled directly from the structure shown in Figure 5. The schema would be compiled from an hierarchical text plan that included the intentions as well as the rhetorical methods and speech acts that are used to achieve the intentions. Figure 5 shows *only* the intentional structure.

If the text produced by the COMPARE & CONTRAST portion of the schema fails to persuade the hearer, the speaker should try other strategies for achieving this intention. For example, the speaker may paraphrase the reasoning that led to recommending this act, cite the advice of experts, or invoke authority. However, because information about intentions has been "compiled out" of the schema representation, the system cannot recover because it cannot determine which goal failed or what other linguistic strategies can be used to achieve the goal.

Note that the problem stems from the fact that, in general, there is not a one-to-one mapping from intentions to schemata or rhetorical predicates. For example, the COMPARE & CONTRAST schema can be used to persuade the hearer to perform a replacement action as in the example above, but it could also be used for several other purposes. This schema could be used to identify the differences between two objects so that the hearer can discriminate one from the other. It could also be used in a strategy where a new concept is defined by comparing it to a known concept. Thus, simply knowing that a COMPARE & CONTRAST schema (or predicate) appears in a text does not tell the system why that COMPARE & CONTRAST appears. As a result, if it fails to achieve its intended effect, the system cannot determine how to recover. To find alternative strategies, the system must know what goal it was trying to achieve!

Rigidity of Schemata. A second problem with schemata is that they are too rigid. In the naturally occurring dialogues we analyzed, we have often observed what appear to be opportunistic effects. One such effect we identified is the tendency of the explainer to define a concept immediately after mentioning that concept in the explanation. One possible reason for this is that the explainer only realizes that the hearer may not know that concept after a sentence including it has been planned or uttered. Recall that this occurred in the teacher–student dialogue in Figure 1 (turn 8). As another example, consider the following explanation given by a doctor when asked about the possible ways to treat migraine (italics indicate our present concern, not spoken emphasis).[4]

> Some of the possible ways that I approach migraine would be, depending on the frequency of your headaches, we would determine which approach I would recommend. [...] So, for example, say that you told me that you had three to four headaches [...] in the month, what I would recommend with that frequency is that you should be on something prophylactically. *Prophylactically basically means preventing the headache from occurring before it actually starts.* If you had infrequent headaches, maybe several times a year, [...] then I would recommend more something abortive. *That means that when the headache came on, I would treat you at that point.* I would rather, to help prevent side effects from you having to take a medicine on a daily basis, just try to abort them, if they were infrequent.

Because a new term can be introduced in virtually any statement, one could only handle this phenomenon in the schema-based approach by incorporating an optional IDENTIFICATION predicate for every term mentioned in the previous predicate after *every* entry in *every* schema. Note that this would be inefficient because if these predicates appear in the schema, the system must consider them (i.e., search for instantiations in

4 This example is taken from transcripts gathered by Claudia Tapia and Johanna Moore at the University of Pittsburgh.

the knowledge base and invoke the focusing mechanism) for *every* additional predicate added to the schema.[5] We believe that a more elegant and efficient approach is to use planning techniques that permit new goals to be posted as they arise and to be worked into an evolving text plan according to rules of discourse as represented in plan operators. The framework that we propose in this paper handles this type of opportunistic planning in a limited way. Suthers (1991) handles a wider array of opportunistic effects using data-driven plan critics to decide when additional information should be included.

3.3 Requirements for a Text Planner

Like others, e.g., Levy (1979) and Appelt (1985), we take the view that speakers have goals to affect the mental states of their hearers and must choose from among the linguistic resources available to them the ones that will satisfy these goal(s). We argued that an approach to text planning that attempts to reason directly about how speech acts can be combined into coherent multisentential texts to achieve a speaker's intentions is likely to be computationally infeasible. However, our discussion of schemata pointed out that we must be careful about what and how much is compiled out of our representations for the sake of efficiency. While we believe the schema-based approach to be correct in spirit, we think that too much information has been lost in schemata composed simply of rhetorical predicates.

Ideally, what we desire is an approach that:

- records enough information about the system's intentions and how those intentions were achieved, so that the system can reason about its own previous utterances to determine how to recover when its explanations are not understood, and
- is capable of producing texts in a computationally efficient manner.

To achieve these goals, we propose an approach to text planning in which plan operators encode knowledge about how intentions may be achieved via a set of techniques commonly found in natural discourse. Thus we require a plan language that links intentions to the rhetorical means for achieving them.

4. Linking Intentions and Rhetorical Structure

Two theories of discourse structure that make the connection between rhetorical relations and speaker intentions have been proposed: Hobbs' (1979, 1983, 1985) theory of coherence relations, and Mann and Thompson's (1988) Rhetorical Structure Theory. As we will see, Rhetorical Structure Theory can be adapted to a computational model in a fairly natural way, and in fact there is an implemented prototype of the theory (Hovy 1991). However, the straightforward operationalization that is used in this prototype suffers from a fundamental problem for our purposes. After discussing the two theories that link intention to rhetorical structure, we discuss this problem in detail. In Section 5, we describe how this problem may be solved and present a text planner that implements this solution.

5 It is also unclear how we could represent this opportunistic strategy in the schema-based approach, since it is only when "unpacking" complex concepts such as `assign-value-to-generalized-variable` that the system recognizes that it will introduce the terms `assign`, `value` and `generalized-variable`. See Moore and Paris (1992) for more discussion about this topic.

4.1 Theoretical Background

Hobbs characterizes coherence in terms of a set of binary **coherence relations** between a current utterance and the preceding discourse. He identified four reasons why a speaker breaks a discourse into more than one clause and classified the relations accordingly. For example, if a speaker needs to connect new information with what is already known by the hearer, the speaker chooses one of the **linkage relations**, such as BACKGROUND or EXPLANATION. As another example, when a speaker wishes to move between specific and general statements, he or she must employ one of the **expansion relations**, such as ELABORATION or GENERALIZATION. According to Hobbs, how the speaker chooses to continue a discourse is equivalent to deciding which relation to employ. From the hearer's perspective, understanding why the speaker continued as he or she did is equivalent to determining what relation was used.

Hobbs originally proposed coherence relations as a way of solving some of the problems in interpreting discourse, e.g., anaphora resolution (Hobbs 1979). He defines coherence relations in terms of inferences that can be drawn from the propositions asserted in the items being related. For example, Hobbs (1985, p. 25) defines ELABORATION as follows:

> ELABORATION: S_1 is an ELABORATION of S_0 if the hearer can infer the same proposition P from the assertions of S_0 and S_1.

Here S_1 represents the current clause or larger segment of discourse, and S_0 an immediately preceding segment. S_1 usually adds crucial information, but this is not part of the definition, since Hobbs wishes to include pure repetitions under ELABORATION.

Hobbs' theory of coherence is attractive because it relates rhetorical relations to the functions that speakers wish to accomplish in a discourse. Thus, Hobbs' theory could potentially tell a text planner what kind of rhetorical relation should be used to achieve a particular goal of the speaker. For example, Hobbs (1979) notes two functions of ELABORATION. One is to overcome misunderstanding or lack of understanding, and another is to "enrich the understanding of the listener by expressing the same thought from a different perspective." However, note that such specifications of the speaker's intentions are not an explicit part of the formal definition of the relation. For this reason we have chosen an alternative theory of text structure, Rhetorical Structure Theory (RST) (Mann and Thompson 1988), as a basis for our text planning system.

In contrast to Hobbs' coherence relations, the definition of each **rhetorical relation** in RST indicates constraints on the two entities being related as well as constraints on their combination, *and a specification of the effect that the speaker is attempting to achieve on the hearer's beliefs or inclinations*. Thus RST provides an explicit connection between the speaker's intention and the rhetorical means used to achieve those intentions. As an example, consider the RST definition of the MOTIVATION relation shown in Figure 6. As shown, an RST relation has two parts: a **nucleus (N)** and a **satellite (S)**.[6] The MOTIVATION relation associates text expressing the speaker's desire that the hearer perform an action (the nucleus) with material intended to increase the hearer's desire to perform the action (the satellite). For example, in the text below, (1) and (2) are related by MOTIVATION:

> (1) Come to the party for the new President. (2) There will be lots of good food.

6 This is an oversimplification. In fact, there are a small number of RST relations, e.g., SEQUENCE and JOINT, that are **multinuclear** and can relate more than two pieces of text.

relation name: MOTIVATION
constraints on N: Presents an action (unrealized with respect to *N*) in which the Hearer is the actor.
constraints on S: none
constraints on N + S combination: Comprehending S increases the Hearer's desire to perform the action presented in N.
effect: The Hearer's desire to perform the action presented in N is increased.

Figure 6
RST relation—MOTIVATION.

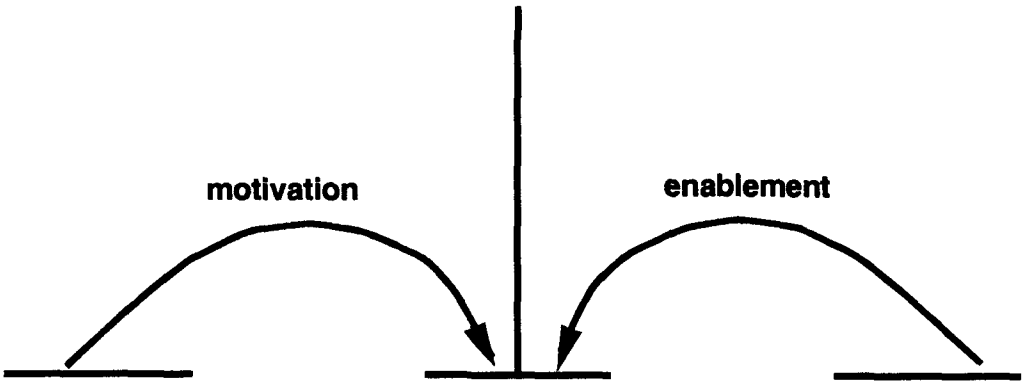

Figure 7
Graphical representation of an RST schema.

The nucleus of the relation is that item in the pair that is most essential to the writer's purpose. In the example above, assuming that the writer's intent is to make the hearer go to the party, clause (1) is nuclear. In general, the nucleus could stand on its own, but the satellite would be considered a non sequitur without its corresponding nucleus. In our example, without the recommendation to *come to the party* the satellite in (2) is out of place. Moreover, RST states that the satellite portion of a text may be replaced without significantly altering the intended function of the text. The same is not true for the nucleus. For example, replacing (2) above with:

(2′) All the important people will be there.

does not greatly change the function of the text as a whole. However, replacing the recommendation in the nucleus, e.g.,

(1′) Don't go to the party.

significantly alters the purpose of the text.

RST relations may be combined into **schemata** that define how a text can be broken down into smaller units. Each schema contains a nucleus and zero or more satellites related to the nucleus by one of the RST relations. RST schemata do not constrain the

ordering of the nucleus and satellites, and relations may occur any number of times within a schema. Furthermore, the schemata are recursive; text serving as a satellite in one schema may itself consist of a nucleus and any number of satellites. A graphical depiction of one schema defined by Mann and Thompson (1988) appears in Figure 7. This schema consists of a nucleus and two satellites: one providing MOTIVATION for the material in the nucleus, and the other providing ENABLEMENT for the material in the nucleus.

4.2 Using RST in Text Generation

Although originally intended for generation, until recently RST was used primarily as a tool for analyzing texts in order to investigate various linguistic issues. The analyst breaks the text down into parts called **text spans** and then tries to find an RST relation that connects each pair of spans until all pairs are accounted for. To determine whether or not a relation holds between two spans of text, the analyst checks to see whether the constraints on the nucleus and satellite hold and whether it is plausible that the writer desires the condition specified in the Effect field. All of the text is given. One need only determine whether or not the constraints are satisfied.

Using RST in a constructive process, such as generating a response to a question, is a very different task. For example, in order to produce text that will succeed in getting the hearer to perform an action, a system must determine what (if any) information in its knowledge base could be used to increase the hearer's desire to perform the action (MOTIVATION), what information could be used to increase the hearer's ability to perform the action (ENABLEMENT), how much of this information to present, in what order, at what level of detail, etc. Moreover, the theory states that the nucleus and satellite portions of a text may occur in any order, relations may occur any number of times, and a nucleus or satellite may be expanded into a text employing any other relation at any point. In order to use RST, a text generation system must have control strategies that dictate how to find such knowledge in the knowledge base, when and what relations should occur, how many times, and in what order.

Mann (1984) suggested that goal pursuit methods used in artificial intelligence could be applied to RST for text generation. Schemata can be viewed as means for achieving the goals stated as their effects, and the constraints on relations as a kind of precondition to using a particular schema. However, much work must be done to formalize the constraints and effects of the RST relations and schemata in order to use RST in a text generation system.

One attempt at formalization was made by Hovy (1991), who operationalized a subset of the RST relation definitions for use as plan operators in a text structuring process. Hovy's structurer employs a top-down planning mechanism to order a given set of input elements into a coherent text. To form plan operators from RST relation definitions, Hovy maps the intended effect of the relation to the Results field of the corresponding operator. In Hovy's system, the contents of the Results field are viewed as the communicative goal(s) that may be achieved by using the associated relation, and the relation name as the rhetorical strategy that achieves the goal(s). The constraints on the nucleus, satellite, and their combination that are specified in the relation definition become subgoals in Hovy's operators. The relation name is also included in the plan operator (and in the evolving plan) so that appropriate connectives can be inserted in the final text.

Figure 8 shows the RST relation definition for the CIRCUMSTANCE relation, and Figure 9 shows Hovy's characterization of this relation as a plan operator. Note from this figure that Hovy's operator includes fields called **growth points**. These post optional subgoals, which will be expanded if information satisfying these goals appears in the

relation name: CIRCUMSTANCE
constraints on N: none
constraints on S: presents a situation (not unrealized)
constraints on N + S combination:
S sets a framework in the subject matter within which Hearer is intended to interpret the situation presented in N.
effect: The Hearer recognizes that the situation presented in S provides the framework for interpreting N.

Figure 8
RST relation—CIRCUMSTANCE.

Name: CIRCUMSTANCE

Results:
((BMB SPEAKER HEARER (CIRCUMSTANCE-OF ?X ?CIRC)))

Nucleus + Satellite requirements/subgoals:
((OR (BMB SPEAKER HEARER (HEADING.R ?X ?CIRC))
(BMB SPEAKER HEARER (TIME.R ?X ?CIRC))))

Nucleus requirements/subgoals:
((BMB SPEAKER HEARER (TOPIC ?X)))

Nucleus growth points:
(BMB SPEAKER HEARER (ATTRIBUTE-OF ?X ?ATT))

Satellite requirements:
((BMB SPEAKER HEARER (TOPIC ?CIRC)))

Satellite growth points:
((BMB SPEAKER HEARER (ATTRIBUTE-OF ?CIRC ?VAL)))

Order: (NUCLEUS SATELLITE)

Relation phrases: (" ")

Figure 9
Hovy's RST plan for CIRCUMSTANCE (from Hovy [1991] p. 90).

set of input items to be expressed. As Hovy (1991, p. 94) points out, RST operators with growth points can be viewed as schemata, and this is how they are used in his implementation. He also argues that by viewing growth points as "suggestions" rather than "injunctions," the RST operators can be used in a more "open-ended" approach to planning. Hovy (1991, p. 98) goes on to suggest a range of criteria that a planner might use to determine when additional material should be included/excluded, but these have not been implemented.

The operator in Figure 9 says that in order to achieve the state where the speaker believes that the hearer and speaker mutually believe that ?CIRC is a circumstance of some event ?X, the speaker can state ?X (this will be the result of posting the nucleus subgoal (BMB SPEAKER HEARER (TOPIC ?X))), and then state the circumstantial information (this will be the result of posting the satellite subgoal (BMB SPEAKER HEARER (TOPIC ?CIRC))). The constraints on the Nucleus + Satellite check that whatever is bound to the variable ?CIRC either stands in a HEADING.R or TIME.R relation to the event bound to ?X.[7] Using this operator, Hovy's structurer can generate portions of

7 This particular operator was used in a naval application where the main events were ship movements

text such as:

Knox is en route to Sasebo. Knox is heading SSW.

The second sentence (the satellite) stands in a CIRCUMSTANCE relation to the first sentence (the nucleus). Typically, this text would be only a portion of a larger paragraph.

4.3 A Problem with the Straightforward Operationalization of RST

Above we have argued that, in order for a system to be able to participate in dialogues, it must have an explicit representation of the intentional structure of its own utterances. Further, we have seen that RST provides a link between intentions and rhetorical relations, and that RST can be adapted in a straightforward manner for use in a text-structuring task by encoding the specification of the intended effect of an RST relation as the goal that the plan operator can be used to achieve, and the constraints on relations as the subgoals that must be satisfied.

However, in our efforts to use RST to construct a text plan that includes both the intentions of individual segments of the text and an indication of the rhetorical relations between segments, we found such an operationalization to be inadequate in the general case. More specifically, we found that there is an important distinction between two previously identified classes of RST relations that must be taken into account when attempting to use RST for text planning. Mann and Thompson (1988, pp. 256–257) break the RST relations into two classes: **presentational** and **subject matter**. Presentational relations are those whose intended effect is to increase some inclination in the hearer, such as the desire to act (MOTIVATION), or the degree of positive regard for (ANTITHESIS), the degree of belief in (EVIDENCE), or the degree of acceptance of (JUSTIFY) the information presented in the nucleus. In contrast, the intended effect of a subject matter relation is that the hearer recognize that the relation in question holds "in the subject matter." For example, VOLITIONAL-CAUSE relates two text spans if the speaker intends the hearer to recognize that the situation presented in the satellite is a cause for the volitional action presented in the nucleus.

The Effects of the presentational relations can be adopted in a straightforward manner as intentions for plan operators. However, the Effects of the subject matter relations are not sufficient for representing speakers' intentions. The Effects of subject matter relations capture the speaker's intention to make the hearer understand a piece of subject matter, but do not indicate *why* the speaker is presenting this information. Consider again the CIRCUMSTANCE relation in Figure 8. The specified effect of this relation is that the hearer knows some circumstance of the situation presented in the nucleus. As we have seen from Hovy's work, this relation is typically used to provide information about the time of an event, the location of an object, etc. But in order to participate in a dialogue, the system must know more than the fact that it intended to convey the time of an event or location of an object: it must know *why* it intended to convey that. We illustrate the problem with an example.

Consider the hypothetical fragment of task-oriented dialogue shown in Figure 10, in which a system is instructing a user in how to take apart a physical device. The system tells the user to remove the cover. In order to increase the user's ability to perform the act, the system employs an ENABLEMENT relation that tells the user what tool to use. Then, *in order to help the user find the appropriate tool*, the system tells the user where the tool is located, a CIRCUMSTANCE. Now, suppose that we have a text planner

and the circumstances that were important to report were heading and time. The operator could be generalized to a wider range of circumstances.

SYSTEM Remove the cover. You'll need the Phillips screwdriver. It's in the top drawer of the toolbox. Do you have it? [1]

USER No. [2]

Figure 10
Fragment of hypothetical task-oriented dialogue.

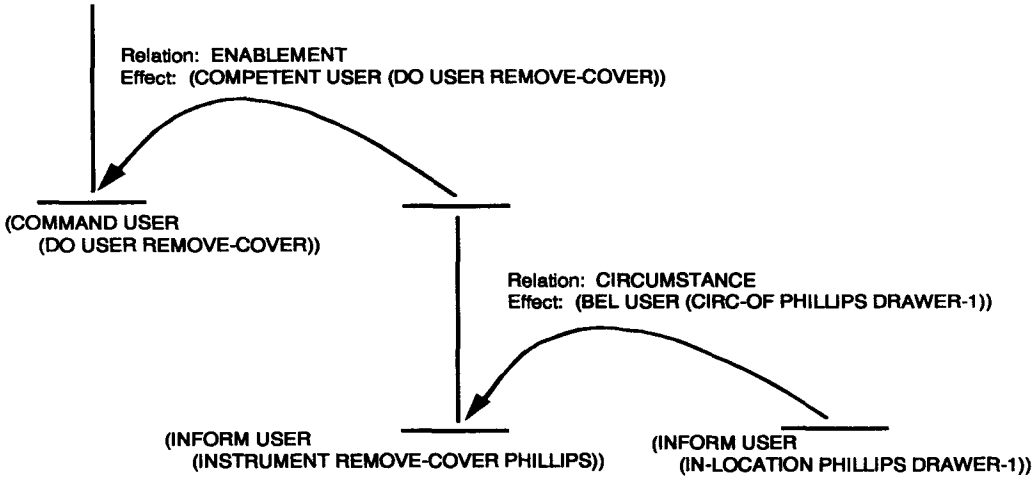

Figure 11
RST tree for system's utterance.

that only records the RST relations used and the speech acts they relate, as shown in Figure 11. Since each RST relation has a relation name as well as an `Effect`, these have both been recorded on the arcs representing the relations.

Now, let us consider how to respond to the user's "No" in the sample dialogue. Clearly the user is indicating an inability to locate the Phillips screwdriver, i.e., either she or he is unable to identify the referent of the term "Phillips screwdriver," or there is no Phillips screwdriver in the toolbox. For the sake of this example, let us assume that the hearer is unable to identify the referent. In terms of building a system that can participate in such dialogues, there are two problems with the representation shown in Figure 11. First, as the user indicates that he or she cannot identify the Phillips screwdriver, it is clear that the portion of the text that failed is the part that attempts to uniquely identify the Phillips screwdriver, namely the text generated in the `CIRCUMSTANCE` satellite: *It's in the top drawer of the toolbox.* However, it is difficult for a system to determine that this is the portion of text that failed, since the only intention represented is that the system wants the hearer to know that the screwdriver being in the toolbox is a circumstance of the Phillips screwdriver. The system has no representation of *why* this bit of circumstantial information is being conveyed.

Second, even if the system could identify this as the offending portion of the plan tree, it is very limited in terms of recovery strategies. In this case, the system's only recourse is to try to find different ways of achieving the subgoal `(BEL USER (CIRCUMSTANCE-OF PHILLIPS ?CIRC))`, i.e., other operators that have this effect but post different subgoals, or by finding different pieces of information that satisfy the constraints on the operator of Figure 9, i.e., alternative bindings for the variable `?CIRC`,

which is currently bound to (IN-LOCATION PHILLIPS DRAWER-1). But it is likely that the location of the Phillips screwdriver is the only piece of circumstantial information relevant to identifying the screwdriver.[8] So presenting other circumstantial information that may be stored in the knowledge base (e.g., when it was purchased, or how long it has been in the toolbox) is almost certainly not relevant and should not be mentioned here. But how can the system tell what is relevant without knowing *why* circumstantial information is being presented? What is missing from the representation in Figure 11 is a critical piece of information indicating that the speaker is presenting the circumstantial information because she or he intends this information to allow the hearer to identify the tool. (We could denote this intention as (KNOW USER (REF PHILLIPS))).

One may argue that the system could recover by trying other operators that implement the ENABLEMENT relation. But again, there are two problems. First, there may not be any other such operators whose constraints are satisfied in this situation. That is, perhaps the only way to remove the cover without damaging it is to use a Phillips screwdriver. Second, even if there were other operators for implementing the ENABLEMENT relation, the system is not behaving intelligently. It will never be able to figure out that it can recover by telling the user some attributive information about the object it is trying to identify, e.g., *The Phillips screwdriver has a yellow handle and a cone-shaped head.* This is because such a text would be produced by an ELABORATION-OBJECT-ATTRIBUTE relation, not a CIRCUMSTANCE relation. Again, this problem arises because the representation lacks the crucial piece of intentional information: (KNOW USER (REF PHILLIPS)), as well as the information that once the Phillips screwdriver has been mentioned, the rhetorical continuations that can be used to help achieve the intention (KNOW USER (REF PHILLIPS)) are CIRCUMSTANCE, ELABORATION-OBJECT-ATTRIBUTE, CONTRAST (one could identify an object by contrasting it with an object known to the user), etc.

The argument ultimately hinges on the observation that, in the case of the subject matter RST relations, the mapping between intentions and rhetorical relations is *not* a one-to-one mapping. In fact, the mapping is many-to-many. This is similar to the argument we made earlier with respect to schemata of rhetorical predicates. Because the mapping of intentions to rhetorical predicates is not one-to-one, intentions cannot be recovered from a record of the rhetorical predicates used to produce a text.

With respect to subject matter relations, we have just shown that an intention such as (KNOW USER (REF PHILLIPS)) may be achieved by a variety of different rhetorical relations. Similarly, a given subject matter relation may be used in service of several different intentions. For example, the CIRCUMSTANCE relation can be used when the speaker wants the hearer to identify the referent of a description (*"It's in the toolbox."*) or know a precondition for an action ("You want flight 101. *It leaves at 8:00 pm."*) In addition, the CIRCUMSTANCE relation is used when the speaker wants the hearer to know how entities are temporally or spatially related. ("He volunteered as a classical music announcer. *That was in 1970.* He left to go to graduate school. *That would've been 1973."*)

Contrast this with the presentational RST relations where the mapping between intention and rhetorical relation *is* a one-to-one mapping. For example, if the speaker's goal is to "increase the hearer's desire to perform the action presented in the nucleus," then whatever text is used to achieve this goal, it stands in a MOTIVATION relation to the nucleus.

8 This is because the knowledge base search originally retrieved this as the information that is most relevant to helping this hearer identify the Phillips screwdriver.

Because of this dichotomy between the two classes of RST relations, we conclude that any approach to discourse structure that relies solely on rhetorical relations or predicates and does not explicitly encode information about intentions is inadequate for handling dialogues. Hovy's (1991) approach suffers from this problem.[9] Moreover, as Moore and Pollack (1992) argue, a straightforward approach to revising such an operationalization of RST by modifying subject matter operators to indicate associated intentions cannot succeed. Such an approach "presumes a one-to-one mapping between the ways in which information can be related and the ways in which intentions combine into a coherent plan to affect a hearer's mental state." We have just shown examples indicating that no such mapping exists.

5. A Text Planner for Advisory Dialogues

In this section we present a text planner that constructs explanations based on the intentions of the speaker at each step and the linguistic means available for realizing these intentions. Given a goal representing the speaker's intention, our planner finds the linguistic resources available for achieving that goal. These linguistic resources can be either speech acts, which are directly satisfiable, or rhetorical strategies indicating how what can be said next relates to what has already been said. While planning, our system records both the intentions behind text spans and the rhetorical relation that holds between each two text spans.

This approach improves upon Hovy's work in two ways. First, as we have seen, Hovy's operationalization of RST relations into plan operators conflates intentional and rhetorical structure. In contrast, our plan language preserves an explicit representation of both intentional and rhetorical knowledge, and thus is more suitable for participating in dialogues. Second, our text planning system is able to select the content to include in its explanations in addition to structuring that content. Recall that Hovy's system is given a set of input elements to express. Like Appelt (1985, pp. 6–10), we believe that the tasks of choosing what to say and choosing a strategy for saying it cannot be divided. They are intertwined and influence one another in many ways. What knowledge is included in a response greatly depends on the speaker's intention and the linguistic strategy chosen to achieve it. For example, the information to be included when describing an object by drawing an analogy with a similar object will be quite different from the information to be included in describing the object by discussing its components. At the same time, which strategy is chosen to satisfy an intention must depend on what knowledge is available. For example, whether the system chooses to draw an analogy will depend on whether an analogous concept familiar to the hearer is available in the system's knowledge sources. In our text planner, decisions are made locally each time alternative strategies for achieving a (sub)goal are considered. The content of the text is not selected a priori.

5.1 The Plan Language

To enable our planner to construct text plans that explicitly capture the intentional and rhetorical structures of the text it produces, we distinguish two types of goals:

- **Communicative goals.** These represent the speaker's intentions to affect the beliefs or goals of the hearer. They are denoted as states, such as the

9 To be fair, the goal of Hovy's work was to show that RST could be used to order a set of input propositions into a coherent text. He did not set out to provide a system that could participate in a dialogue.

state in which the hearer believes a certain proposition, has a goal to perform an action, or has the knowledge necessary to perform an action. The presence of a communicative goal in a text plan does not cause any text to be generated directly. Achieving goals of this type leads to the posting of linguistic goals.

- **Linguistic goals**. These correspond to the linguistic means available to speakers for achieving their communicative goals. They lead to the generation of text and are of two types: **speech acts** and **rhetorical goals**. In our system, we assume that speech acts, such as `INFORM` or `RECOMMEND`, can be straightforwardly mapped into utterances that form part of the final text.[10] They are considered primitives by the text planner. Rhetorical goals, such as `MOTIVATION` and `CIRCUMSTANCE`, cannot be achieved directly and must be refined into one or more subgoals. The strategies for achieving them may post further communicative goals or speech act goals in order to express satellite information in a text. They are intended to establish rhetorical links between text spans, and often cause connective markers to be generated in the final text.

We distinguish these two types of goals for two reasons. First, the distinction is necessary to handle the many-to-many mapping between intentions and rhetorical strategies for achieving them. Table 1 summarizes the mapping between intentions and rhetorical relations we have identified in generating the texts necessary for our application. These mappings have been encoded in our library of plan operators. In this table, the presentational RST relations appear above the double line. As shown, there *is* a one-to-one mapping between these relations and speaker intentions. However, note that for the subject-matter relations that appear below the double line, there is *not* a one-to-one mapping between rhetorical relations and speaker's intentions. In general, there may be many different rhetorical strategies for achieving any given intention. For example, as discussed above and indicated in Table 1, the intention `(KNOW ?h (REF ?object-descriptor))` can be achieved by telling the hearer circumstantial information about the object (`CIRCUMSTANCE`), by contrasting the object with an object known to the user (`CONTRAST`), or by telling the hearer some of the attributes or parts of the object (`ELABORATION-OBJECT-ATTRIBUTE` or `ELABORATION-WHOLE-PART` respectively). Moreover, the mapping in Table 1 makes it clear that a particular rhetorical strategy may serve many intentions. For example, `CONTRAST` may be used to identify the referent of a description, to define a new term, to identify the differences between entities, to make the hearer believe that a method is the best one for achieving a domain goal, etc. As we will see, operators in our plan language achieve intentions by posting rhetorical subgoals and/or speech acts.

The second reason for separately and explicitly representing the two types of goals is so that the completed plan structure will contain an explicit representation of the speaker's intentions as well as a record of the speech acts and rhetorical strategies used to achieve them. As we have argued above, it is essential that the intentional structure of a text be recorded so that the system may respond to the user's follow-up questions. In addition, having the rhetorical structure explicitly noted in the text plan allows the text generator to include discourse markers in the final text in a straightforward manner. Such markers enhance the coherence of the text and aid the reader in understanding the text as a whole, by helping him or her understand how

10 Here we are using the term "speech act" where Appelt (1985) would use "surface speech act."

Table 1
Intention to rhetorical relation mapping.

Intention	Rhetorical Relation	
(PERSUADED ?h ?proposition) (PERSUADED ?h (DO ?h ?act)) (COMPETENT ?h (COMPREHEND ?h ?x)) (COMPETENT ?h (DO ?h ?act))	EVIDENCE MOTIVATION BACKGROUND ENABLEMENT	**Presentational Relations**
(KNOW ?h (REF ?desc))	CIRCUMSTANCE CONTRAST ELABORATION-OBJECT-ATTRIBUTE ELABORATION-WHOLE-PART	**Subject Matter Relations**
(KNOW-ABOUT ?h ?concept)	CIRCUMSTANCE CONDITION CONTRAST ELABORATION-GENERAL-SPECIFIC ELABORATION-OBJECT-ATTRIBUTE ELABORATION-SET-MEMBER ELABORATION-SPECIFIC-GENERAL ELABORATION-WHOLE-PART PURPOSE SEQUENCE	
(BEL ?h (REF (DIFFS ?x ?y) ?d))	CONTRAST	
(BEL ?h (STEP ?act ?goal))	ELABORATION-GENERAL-SPECIFIC ELABORATION-PROCESS-STEP SEQUENCE	
(BEL ?h ?proposition)	CONTRAST ELABORATION (all types)	
(BEL ?h (METHOD-FOR ?goal ?method))	CONDITION CONTRAST MEANS SEQUENCE	
(BEL ?h (BEST-METHOD-FOR ?goal ?method))	CONCESSION CONDITION CONTRAST OTHERWISE	

the parts of the text relate to one another (Brewer 1980; Cahour, Falzon, and Robert 1990; Ehrlich and Cahour 1991; Goldman and Durán 1988; Levy 1979; Meyer, Brandt, and Bluth 1980).[11] In a system such as ours, sets of connectives are associated with each rhetorical relation, and one can be chosen based on features of the final text being produced (e.g., whether or not a sentence boundary occurs between nucleus and satellite, etc.). The system need not reason directly about the relationships between the effects of speech acts to determine a suitable connective—a process we believe to be much more computationally intensive.

5.2 Representing Plan Operators

Our plan language provides operators for achieving the two types of goals presented in the previous section. Each operator consists of:

- **an effect:** a characterization of the goal that this operator can be used to achieve. This may be a communicative goal, such as *The speaker intends to*

11 Note that rhetorical structure is not the only source of discourse markers. They may be used to mark shifts in attentional structure, discourse segment boundaries, or aspects of the exchange structure in interactive discourse (Grosz and Sidner 1986; Redeker 1990; Schiffrin 1987).

achieve the state in which the hearer believes a proposition or a linguistic goal, such as *Establish motivation between an act and a goal* or *Inform the user of a proposition.*

- **a constraint list:** a list of conditions that should be true in order for the operator to have the intended effect. Constraints may refer to facts in the system's domain knowledge base, information in the user model, information in the dialogue history, or information about the evolving text plan.
- **a nucleus:** a subgoal that is most essential to achievement of the operator's effect. Every operator must contain a nucleus.
- **satellites:** additional subgoal(s) that may contribute to achieving the effect of the operator. An operator can have zero or more satellites. When present, satellites may be required or optional. Unless otherwise indicated, a satellite is assumed to be required.

5.2.1 Representing Mental States. Before providing examples of operators encoded in the plan language, we introduce the representational primitives used to express the knowledge contained in plan operators. The 12 predicates listed here were sufficient to represent the communicative goals that were needed to produce responses to the range of questions handled by the PEA system. Development of other application systems in a range of domains using the text planner described here is underway, and experience with these systems will help us determine whether this set of mental states is sufficient or must be extended (Carenini and Moore 1993; Rosenblum and Moore 1993).

We express information about agents' mental states in the following terms. In the notation below, constants are denoted by symbols written in all uppercase letters, e.g., `DO`, and typed variables are denoted by lowercase symbols beginning with a "?", e.g., `?agent`.

1. `(KNOW-ABOUT ?agent (CONCEPT ?c))`: The agent knows of the existence of the concept `?c`. This does not imply that the agent knows any particular properties of the concept, its subconcepts, the instances of this concept, or how this concept is used in problem solving.
2. `(KNOW ?agent (REF ?description))`: The agent can identify the real world entity described by `?description`.
3. `(BEL ?agent (?predicate ?e1 ?e2))`: The agent believes that the two-place predicate holds between entities `?e1` and `?e2`.
4. `(BEL ?agent (SOMEREF (?predicate ?arg1 ?arg2 ...?argN-1)))`: The agent believes that there exists some entity(ies) satisfying the N-ary predicate, i.e., there is some referent of this N-ary predicate.
5. `(BEL ?agent (REF (?predicate ?arg1 ?arg2 ...?argN-1) ?referent))`: The agent believes that the referent satisfies the N-ary predicate. This notation is used when $N > 2$. When $N = 2$, we use the notation shown in 3 above.
6. `(GOAL ?agent ?goal)`: The agent has adopted the specified domain goal. In this expression, `?goal` must be a nonprimitive domain action, i.e., a goal that requires further refinement before it can be achieved.

7. `(GOAL ?agent (DO ?agent ?action))`: `?agent` has adopted a goal to perform the specified action. The action must be a primitive in the domain.
8. `(PERSUADED ?agent (DO ?agent ?act))`: The agent is persuaded to perform the action at some unspecified time in the future. This is how we have chosen to represent the RST effect *Increase the hearer's desire* to perform an action.
9. `(PERSUADED ?agent (ACHIEVE ?agent ?goal))`: The agent is persuaded to adopt the goal.
10. `(PERSUADED ?agent ?proposition)`: The agent is persuaded that `proposition` is true. This is how we have chosen to represent the RST effect *Increase the hearer's belief* in a proposition.
11. `(COMPETENT ?agent (DO ?agent ?act))`: The agent knows the information necessary to perform the primitive domain action.
12. `(COMPETENT ?agent (ACHIEVE ?agent ?goal))`: The agent knows a method for achieving the nonprimitive domain action ?goal.

Representing Communicative and Linguistic Goals. Communicative goals represent the speaker's intention to produce a certain effect on the hearer's mental state, e.g., to make the hearer believe some proposition or adopt a goal to perform a certain action. In the plan language, communicative goals are written simply in terms of mental states of the hearer. When a goal of the form `(BEL ?hearer ?proposition)` appears in a text plan, it should be read as *The speaker intends to achieve the state where ?hearer believes ?proposition*. When `(BEL ?hearer ?proposition)` appears in the effect field of an operator, it indicates that the operator is capable of having this effect on the hearer's mental state.

To achieve a communicative goal, operators post linguistic (rhetorical and speech act) subgoals. Rhetorical goals are of the form `(relation-name arg1 ...argN)`, where `relation-name` is one of the relations defined in RST. An expression of this form represents a goal to establish the rhetorical relation between the entities listed as arguments. For example, `(MOTIVATION REPLACE-SETQ-WITH-SETF ENHANCE-READABILITY)` indicates the speaker's rhetorical goal to establish that the domain goal `ENHANCE-READABILITY` is motivation for the act `REPLACE-SETQ-WITH-SETF`.

Communicative and rhetorical goals are eventually refined into primitive actions that can be executed to cause changes in the hearer's mental state. In our planning formalism, we treat speech acts as primitive. Speech act goals thus appear at the leaf nodes of text plans. There are currently four speech acts in our system: `INFORM`, `ASK`, `RECOMMEND`, and `COMMAND`.

Finally, there are two special forms in the plan language: `FORALL` and `SETQ`. These may appear in the nucleus or satellite fields of plan operators. `FORALL` has the form

```
(FORALL ?variable-name ?goal).
```

A `FORALL` clause in a nucleus or satellite field causes an instance of the goal, `?goal`, to be posted for each possible binding of the variable named by `?variable-name`. `?goal` may be a communicative goal, a rhetorical goal, or a speech act. Its specification must contain an occurrence of the variable named by `?variable-name`. An example of this special form appears in the nucleus of the plan operator in Figure 14.

The SETQ special form is expressed as:

```
(SETQ ?variable-name ?expression).
```

SETQ causes the variable named ?variable-name to be bound to the result of evaluating the expression ?expression. This special form is useful for assigning values to variables that will be used in later steps of the nucleus or satellites. An example appears in the operator in Figure 16.

Representing Operator Constraints. Constraints on the user's knowledge and goals are expressed using the notation given above for representing the hearer's mental states. Any of the expressions described above can appear in the constraint list of an operator. For example, if the expression

```
(KNOW-ABOUT USER (CONCEPT STORAGE-LOCATION))
```

appears in the constraint field of an operator, it indicates that in order to use the operator (without making assumptions), the hearer should be familiar with the concept STORAGE-LOCATION.

Constraints on the expert system's knowledge are of the form (?predicate ?arg1 ...?arg-N), where ?predicate is an N-ary predicate referring to the expert system's domain knowledge. Since the expert system's knowledge base is made up of several complex data structures, predicates are not tested by a simple unification process. Instead an access function must be written for each predicate in order to test if an instantiated predicate is true, or to find acceptable bindings when a predicate contains variables in some argument positions. The range of predicates over the domain knowledge is quite large, and therefore we will not enumerate them here. We provide English paraphrases of knowledge base constraint predicates wherever they appear in examples.

Finally, constraints may refer to the dialogue history or the status of the current text plan under construction. There are currently two types of constraints in this category. First, there are constraints that indicate whether the operator can be used in nucleus or satellite position in a text plan. The clause (NUCLEUS) in the constraint field of an operator, indicates that this operator can be used to expand the nucleus branch of a text plan. Likewise, the clause (SATELLITE) indicates that the operator can be used to expand a satellite branch.

Second, there are constraints on the focus of attention. We are currently using a simple implementation of local focus rules based on the work of Sidner (1979) and McKeown (1985). We have found the need for two such constraints in the operators we have encoded thus far:

- (CURRENT-FOCUS ?entity): indicates that the operator can be used if ?entity is currently in focus.
- (IN-POTENTIAL-FOCUS-LIST ?entity): indicates that the operator can be used if ?entity is on the list of items that have just been mentioned, and therefore could become the next focus.

5.2.2 Operationalizing RST Schemata. Given these notational conventions, let us consider how we operationalize an RST schema in our plan language. In general, several operators are required to represent the knowledge in an RST schema. For example, Figures 12, 14, and one of Figures 15 or 16 operationalize a portion of the RST schema

In Plan Language Notation:

EFFECT: **(GOAL ?hearer (DO ?hearer ?act))**
CONSTRAINTS: **(NUCLEUS)**
NUCLEUS:
(RECOMMEND ?speaker ?hearer (DO ?hearer ?act))
SATELLITES:
(((PERSUADED ?hearer (DO ?hearer ?act)) *optional*)
((COMPETENT ?hearer (DO ?hearer ?act)) *optional*))

English Paraphrase:

To make the hearer want to do an *act*,
IF this text span is to appear in nucleus position, THEN
1. Recommend the *act*

AND optionally,
2. Achieve state where the hearer is persuaded to do the *act*
3. Achieve state where the hearer is competent to do the *act*

Figure 12
Plan operator for recommending an act.

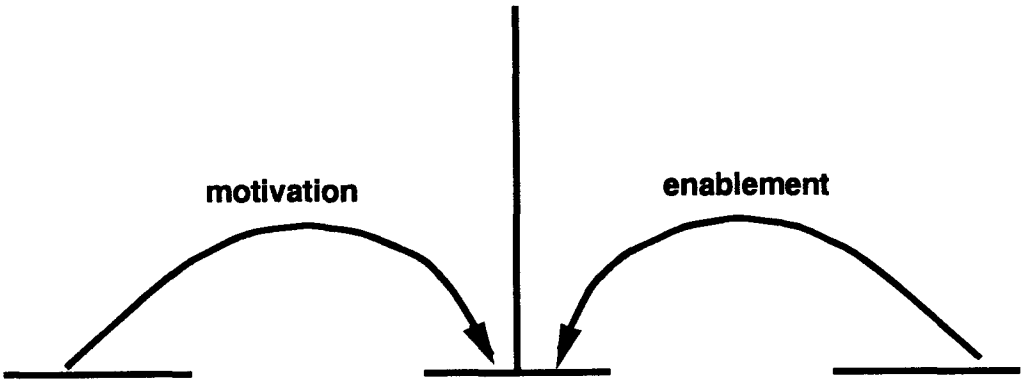

Figure 13
Graphical representation of an RST schema.

depicted in Figure 7, repeated in Figure 13 for the reader's convenience. The operator shown in Figure 12 says that one way of achieving the state where the hearer has the goal of performing an action is to recommend the act (the nucleus), and, optionally, to post a subgoal to achieve the state where the hearer is persuaded to perform the recommended act (the first satellite), and, optionally, to post a subgoal to achieve the state where the hearer has the knowledge necessary to perform the act (the second satellite). `RECOMMEND` is considered a primitive in our system, and can be mapped directly into a specification for the sentence generator. Therefore no operators are needed to achieve it.

However, the system does require operators for achieving the two satellite subgoals. Figure 14 shows an operator that can be used to achieve the first satellite. Informally, this plan operator states that if an act is a step in achieving some goal(s) of

In Plan Language Notation:

EFFECT: **(PERSUADED ?hearer (DO ?hearer ?act))**
CONSTRAINTS: **(AND (STEP ?act ?goal)**
(GOAL ?hearer ?goal)
(MOST-SPECIFIC ?goal)
(CURRENT-FOCUS ?act)
(SATELLITE))
NUCLEUS: **(FORALL ?goal**
(MOTIVATION ?act ?goal))
SATELLITES: **nil**

English Paraphrase:

To achieve the state in which the hearer is persuaded to do an *act*,
IF the *act* is a step in achieving some *goal(s)* of the hearer,
AND the *goal(s)* are the most specific along any refinement path
AND *act* is the current focus of attention
AND the planner is expanding a satellite branch of the text plan
THEN motivate the *act* in terms of those *goal(s)*.

Figure 14
Plan operator for persuading user to do an act.

the hearer, the speaker can persuade the hearer to perform the action by motivating the action in terms of those goals. This plan operator thus indicates that the communicative goal of persuading the hearer to do an act can be achieved by using the rhetorical strategy `MOTIVATION`. This operator thus links the intention with the rhetorical means used to achieve it.

Finally, the system needs an operator that can achieve a `MOTIVATION` subgoal. In our current operator library, there are several such operators. Two of these are shown in Figures 15 and 16. The operator in Figure 15 is very general and can be used to motivate any action in terms of a hearer goal that the act may help to achieve. It posts a subgoal to make the hearer believe that the act is a step in achieving the goal. In the simplest case, this subgoal will be refined directly into a surface speech act that informs the hearer of this fact. The operator in Figure 16 is a more specific operator and can be used only when the act to be motivated is a replacement (e.g., replace `SETQ` with `SETF`). In this case, one strategy for motivating the act is to compare the object being replaced and the object that replaces it with respect to a goal of the hearer.

The three operators shown in Figures 12, 14, and 15 together form one operationalization of a portion of the RST schema shown in Figure 13. Referring back to this schema and the definition of the RST relation `MOTIVATION` shown in Figure 6, the reader will note that these three operators can be used to produce the nucleus and the `MOTIVATION` satellite portion of the schema. To complete the operationalization of the full schema shown in Figure 13, we must also provide operators to refine the second optional satellite, `(COMPETENT ?hearer (DO ?hearer ?act))`. For the sake of brevity, we have omitted these operators here.

There are three important points to note about our operationalization. First, operators like the one shown in Figure 14 provide an explicit link between speaker intentions and the rhetorical means that achieve them. In the case of this particular operator, there is actually a one-to-one mapping between the intention (*Achieve state where*

EFFECT: **(MOTIVATION ?act ?goal)**
CONSTRAINTS: **(AND (STEP ?act ?goal)**
(GOAL ?hearer ?goal))
NUCLEUS: **(BEL ?hearer (STEP ?act ?goal))**
SATELLITES: **nil**

Figure 15
Plan operator for motivating any action by stating the shared goals that act is a step in achieving.

EFFECT: **(MOTIVATION ?act ?goal)**
CONSTRAINTS: **(AND (STEP ?act ?goal)**
(GOAL ?hearer ?goal)
(ISA ?act REPLACE))
NUCLEUS: **((SETQ ?replacee (FILLER-OF OBJECT ?act))**
(SETQ ?replacer (FILLER-OF GENERALIZED-MEANS ?act))
(BEL ?hearer
(SOMEREF (DIFFERENCES-WRT ?replacee ?replacer ?goal))))
SATELLITES: **nil**

Figure 16
Plan operator for motivating a replace act by describing differences between replacer and replacee.

hearer is persuaded to do an act) and the rhetorical strategy for achieving it (`MOTIVATION`). However, as Table 1 shows, this is not the case for subject matter relations. By providing operators that explicitly link intentions with rhetorical strategies, our system can handle the many-to-many mapping between intentions and the communicative strategies that achieve them. So, to return to the Phillips screwdriver example, our system would have several plan operators for achieving the intention: `(KNOW ?hearer (REF ?description))`. One plan operator would indicate that a rhetorical strategy for achieving this goal is to tell the hearer the location of the object (`CIRCUMSTANCE`), another would indicate that another way to achieve the goal is to tell the hearer the identifying attributes of the object (`ELABORATION-OBJECT-ATTRIBUTE`), etc.

Second, operators like the one shown in Figure 14 contribute to the computational efficiency of the system. These operators encode knowledge about the rhetorical strategies that may be used to satisfy particular intentions. Other operators, e.g., the `MOTIVATION` operators shown in Figures 15 and 16, encode different methods for realizing these rhetorical strategies in different situations. Thus we have provided a plan language that preserves an explicit representation of the intention behind each portion of the text, while maintaining the efficiency advantages that originally motivated the use of schemata of rhetorical predicates or relations for natural language generation.

Third, as illustrated by the two alternative operators for achieving a `MOTIVATION` subgoal shown in Figures 15 or 16, in our plan language we can represent very general strategies that are applicable across domains, as well as very specific strategies that may be necessary to handle the idiosyncratic language used in a particular domain. While one may argue that the operator in Figure 16 could be replaced by a more general, domain-independent operator, this does not obviate the need for domain-specific communication strategies. Rambow (1990) argues that domain-specific communication knowledge must be used (whether implicitly or explicitly) in all planned communica-

tion, and advocates that **domain communication knowledge** be represented explicitly. In our plan language, some types of domain-specific communication strategies can be represented in plan operators. When there are multiple operators capable of achieving a given effect, the constraint mechanism controls which operators are deemed appropriate in a given context. Note, however, that we do not wish to claim that a top-down planning formalism that maps speakers' intentions to rhetorical and speech acts can or should be used to generate all text types. In particular, Kittredge, Korelsky, and Rambow (1991) have shown that RST cannot account for the structure of report texts. Moreover, they argue that report generation does not require reasoning about the speaker's intentions to affect the mental attitudes of the hearer. For report generation, they advocate a data-driven approach in which domain communication knowledge plays a central role.

5.3 Constraints Integrate Multiple Sources of Knowledge

Operators contain applicability constraints that specify the knowledge that must be available and the state of the text-planning process that must exist if the operator is to be used. These constraints integrate multiple sources of knowledge; they may refer to the expert system's knowledge bases, the user model, the dialogue history, or the evolving text plan. To our knowledge, our text planner is unique in its explicit representation of constraints from all of these knowledge sources.

It is important to recognize that in our formalism, constraints do more than just limit the applicability of an operator. They also specify the type of knowledge to be included in an explanation, so that the process of satisfying constraints causes the planner to *find* information that will be included in the text. Thus, the selection of information to be presented and the determination of how to present that information are truly integrated in our planning model.

For example, when attempting to apply the operator shown in Figure 14, the variables `?act` and `?hearer` will be bound. What is not yet determined is which domain goals should be used to motivate the act. Checking to see if the first three constraints on this operator are satisfied causes the planner to search its knowledge sources to find acceptable bindings for the variable `?goal`. The first constraint, `(STEP ?act ?goal)`, says that there must be some domain goal(s) that the act is a step in achieving. Satisfying this constraint requires the planner to search the expert system's domain planning knowledge for such goals. The second constraint, `(GOAL ?hearer ?goal)`, further specifies that any such domain goals must be goals of the hearer (user). To check this constraint, the system must inspect the user model. The third constraint requires that `?goal` be the most specific goal along any refinement path satisfying the first two constraints.

The last two constraints of the operator in Figure 14 refer to the evolving text plan. The fourth constraint, `(CURRENT-FOCUS ?act)`, says that this operator can be used when the act is the current focus of attention. The fifth and final constraint, `(SATELLITE)`, says that this operator can only be used when the planner is working on a satellite branch of the current text plan.

In our system, constraints are treated differently depending on the knowledge source to which they refer. We consider the expert system's domain knowledge to be complete and correct. Therefore, constraints referring to the expert system's knowledge bases are considered "rigid." If any one of these constraints fails to be satisfied, the operator is rejected from consideration immediately. In contrast, we do not assume that the system's model of the user is either complete or correct. Thus, constraints referring to the user's knowledge state are treated more loosely. They specify what the user should know in order to understand the text that will be produced by the oper-

ator. However, when selecting an operator, if the user model contains no information relevant to a particular constraint, the planner may simply assume that the constraint is satisfied. When it does so, the assumption is recorded in the plan structure so that this assumption can be questioned if the explanation is not understood.

In the current implementation, constraints on the dialogue history and evolving text plan are rigid. However, we are exploring the idea of treating these as preconditions, since, if they are not true in a given situation, they could be made true by generating some additional text. For example, if a constraint such as `(IN-POTENTIAL-FOCUS-LIST ?act)` is not true, it can be made true by using rhetorical techniques to introduce `?act`, if it is a new topic, or to return to `?act`, if it has been mentioned before. User model constraints can also be treated as preconditions in cases where the system has the underlying knowledge to support explanations that could make them true. We would like to provide our text planner with the ability to choose between making an assumption or planning text to satisfy a user model constraint. Modifying the architecture to support this type of reasoning is relatively straightforward. The more difficult problem is to identify heuristics that guide the text planner in making the most effective choices. When we incorporate the notion of preconditions into our plan operators, we will make a distinction between constraints that the system can try to satisfy and those that must already be satisfied before the operator can be applied. This distinction was first made by Litman and Allen (1987) and later by Maybury (1992).

5.4 The Planning Mechanism

An overview of the explanation generation facility and its relation to other components in the system is shown in Figure 17. The text planner produces a plan for an explanation using the operators in its plan library. The planning process begins when a communicative goal is posted. This may come about in one of two ways. First, in the process of performing a domain task, the expert system may need to communicate with the user, e.g., to ask a question or to recommend that the user perform an action. To do so, it posts a communicative goal to the text planner. Alternatively, the user may request information from the system. In this case, the **query analyzer** interprets the user's question and formulates a communicative goal. Note that a communicative goal such as *Achieve the state where hearer knows about concept c* is really an abstract specification of the response to be produced.

When a goal is posted, the planner identifies all of the potentially applicable operators by searching its library for all operators whose effect field matches the goal. To make this search more efficient, plan operators are stored in a discrimination network based on their effect field. For each operator found, the planner then checks to see if all of its constraints are satisfied. Those operators whose constraints are satisfied (or, in the case of user model constraints, can be assumed to be satisfied) become candidates for achieving the goal.

From the candidates, the planner selects an operator based on several factors, including what the user knows (as indicated in the user model), the conversation that has occurred so far (as indicated in the dialogue history), the relative specificity of the candidate operators, and whether or not each operator requires assumptions to be made. The knowledge of preferences is encoded into a set of selection heuristics. A discussion of the selection process is beyond the scope of this paper; see Moore (in press) for details. Once a plan operator has been selected, it is recorded in the current plan node as the selected operator, and all other candidate plan operators are recorded in the plan node as untried alternatives. If the operator chosen requires any assumptions to be made, they are also recorded in the plan node. The planner then

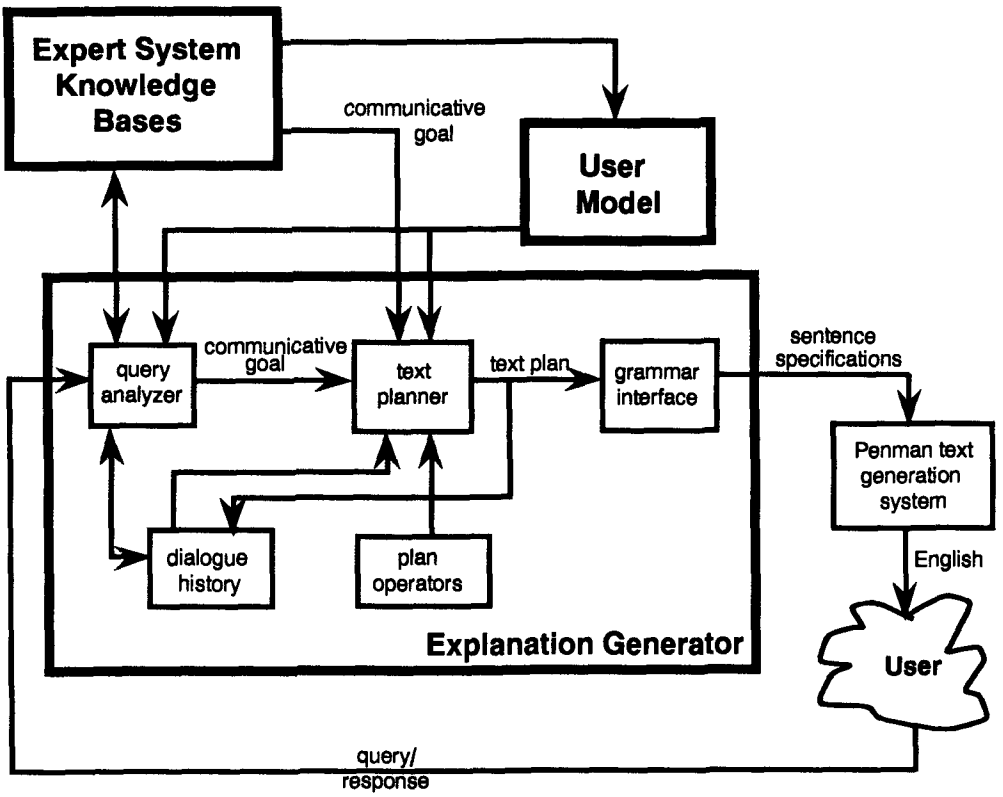

Figure 17
Architecture of explanation system.

expands the selected plan operator by posting its nucleus and required satellites as subgoals to be refined.

Recall that in Section 4.2 we noted that a text generator must decide on the ordering of the text spans it produces. When instantiating an operator, the planner must decide on the ordering of the subgoals in the nucleus and satellite. There are really two ordering issues: (1) for each satellite, the planner must determine whether that satellite should appear before or after the nucleus, and (2) whenever there are multiple subgoals in a set of satellites, the planner must determine the order of these subgoals.

In our current framework, the second type of ordering knowledge is compiled into our text-planning strategies. That is, subgoals should be expanded in the order in which they appear in the plan operator. This is the approach taken by most systems employing schemata, e.g., McKeown (1985); Paris (1991b), and is also common in systems that make use of linear planners, e.g., Cawsey (1993); Hovy (1991); Maybury (1992). Again, this leads to computational efficiency, since the planner does not have to reason about ordering among sibling subgoals. However, some flexibility is lost. We plan to investigate this tradeoff in our future work.

To solve the first ordering problem, we appeal to ordering information provided by RST. Although RST does not strictly constrain ordering, Mann and Thompson (1988) have observed that, for some relations, one ordering is significantly more frequent than

the other. In RST, these ordering observations are treated as "strong tendencies" rather than constraints. For example, in an EVIDENCE relation, the nuclear claim usually precedes the satellite that provides the evidence for the claim; in contrast, in a BACKGROUND relation, the satellite providing the background information necessary to understand the nuclear proposition usually precedes the nucleus. Our planner expands the nucleus before the satellite except for those relations that have been identified as having a typical order of satellite before nucleus.

Finally, the planner must decide when to expand optional satellites. In the current implementation, the planner has two modes, terse and verbose. In terse mode, no optional satellites are expanded. In verbose mode, each satellite is checked against the user model, and those that do not duplicate the user's existing knowledge are expanded. For example, the plan operator in Figure 12 has two satellites. The first satellite calls for persuading the hearer to perform the act. The second satellite calls for making the hearer competent to perform the act and would, if expanded, provide any information the hearer needs to know in order to be capable of performing the act. These satellites are both marked "optional," indicating that it would be sufficient to simply state the recommendation. In verbose mode, the planner will check each of these satellites against the user model. To avoid expanding the first satellite, the user model would have to contain information indicating that the user already has the goal of performing the recommended act. However, note that this would never be the case, since if it were, the system would not be planning text to make the hearer adopt the goal of doing the act. That is, it would not be expanding this operator in the first place. The second satellite provides a more interesting example. In verbose mode, if the user model indicates that the user knows how to perform replacement actions, the second satellite will not be expanded, since the system believes that the user has the knowledge necessary to perform the recommended act.

The planner maintains an agenda of pending goals to be satisfied. When instantiating a plan operator, it creates a new node for the nucleus subgoal and puts it into a list. The planner then expands each of the required satellites and adds them to this list. If the satellite is one that should precede the nucleus, the new satellite node is appended to the front of the list. Otherwise, it is appended to the back of this list. The planner then considers each of the optional satellites. For each of the ones it decides to expand, the planner creates a new node and adds it to the appropriate end of the list. The list is then appended to the *front* of the agenda of goals to be expanded. In this way a text plan is built in depth-first order.

When a speech act is reached, the system constructs a specification that directs the realization component, Penman (Mann and Matthiessen 1985; Penman Natural Language Generation Group 1989), to produce the corresponding English utterance. The system builds these specifications based on the type of speech act, its arguments, and the context in which it occurs.[12] As the planner examines each of the arguments of the speech act, new goals may be posted as a side effect. If one of the arguments is a concept that the user does not know (as indicated in the user model), a satellite subgoal to define this new concept is posted. In addition, if the argument is a complex data structure that cannot be realized directly by English lexical items, the planner will have to "unpack" this complex structure, and, in so doing, will discover additional concepts that must be mentioned. Again, if any of these concepts is not known to the user, subgoals to explain them are posted. An example of this phenomenon is given

12 Bateman and Paris (1989, 1991) are investigating the problem of phrasing utterances for different types of users and situations.

SYSTEM	What characteristics of the program would you like to enhance?	[1]
USER	Readability and maintainability.	[2]
SYSTEM	You should replace (SETQ X 1) with (SETF X 1). SETQ can only be used to assign a value to a simple-variable. In contrast, SETF can be used to assign a value to any generalized-variable. A generalized-variable is a storage location that can be named by any accessor function.	[3]

Figure 18
Sample dialogue.

in the next section. In this way, our system can *opportunistically* define a new term when the need arises. Contrast this approach to the schema-based approach described earlier. Recall that handling this type of phenomenon with schemata would require that the definition of each schema explicitly include an optional `Identification` predicate after *every* entry in *every* schema.

Planning is complete when all subgoals in the text plan have been refined to speech acts. It is important to note that text planning proceeds in such a way that speech acts are planned *in the order in which they will appear in the final text*. This provides two advantages. First, the text plan is a record of the system's utterances in the order in which they are generated. Because our text plans also include the intentional structure of the final text, focus information can be derived from a completed text plan, and there is no need to maintain a separate data structure for managing focus information. Second, by doing the planning in this manner, the planner can easily be extended for incremental generation in which planning and realization are interleaved.

6. Participating in Explanatory Dialogues: An Example

In this section, we provide an example illustrating how our system constructs a text plan for recommending an action. We contrast the text plan produced by our system with the schema representation we showed in Figure 4, and show how the text plan may then be used to handle two follow-up questions. See Moore (in press) for a more detailed discussion of the planning process and additional examples.

Let us return to the sample dialogue with the PEA system that was shown in Figure 3 and that we include again in Figure 18 for the reader's convenience. In this dialogue the user indicates a desire to enhance the readability and maintainability of his or her program. To enhance maintainability, the expert system determines that the user should replace `SETQ` with `SETF`. To recommend this transformation, the expert system posts the communicative goal `(GOAL USER (DO USER REPLACE-1))` to the text planner. This goal says that the speaker would like to achieve the state where the hearer has adopted the goal of performing the act `REPLACE-1`.

A plan operator capable of satisfying this goal was shown in Figure 12. The nucleus is expanded first, causing `(RECOMMEND SYSTEM USER (DO USER REPLACE-1))` to be posted as a subgoal. `RECOMMEND` is a speech act goal that can be achieved directly, and thus expansion of this branch of the plan is complete. Focus information—the current focus and the potential focus list—is also updated at this point. In this case, the current focus is the act `REPLACE-1` and the potential focus list includes the participants in this act, i.e., `USER`, `SETQ`, and `SETF`.

Next, the planner must expand the satellites. Since both satellites are optional in this case, the planner must decide which, if any, are to be posted as subgoals. For the purposes of this example, assume that the planner is in verbose mode and that the user model indicates that the user has the knowledge necessary to perform replacement

acts (i.e., he or she knows how to use the text editor). Thus, only the first satellite will be expanded, posting the communicative subgoal to achieve the state where the user is persuaded to perform the replacement, i.e., (PERSUADED USER (DO USER REPLACE-1)). A plan operator for achieving this goal using the rhetorical relation MOTIVATION was shown in Figure 14.

When attempting to satisfy the constraints of the operator in Figure 14, the system first checks the constraint (STEP REPLACE-1 ?goal). This constraint states that, in order to use this operator, the system must find a domain goal, ?goal, which REPLACE-1 is a step in achieving. To find such goals, the planner searches the expert system's problem-solving knowledge. A detailed discussion of the expert system framework we use is beyond the scope of this paper. However, it is important to note that the type of explanation capability we are describing in this paper places stringent requirements on the way domain knowledge is represented and used in reasoning. Interested readers may find a thorough treatment of this topic in Swartout (1983), Clancey (1983), Neches, Swartout, and Moore (1985), Swartout, Paris, and Moore (1991), and Moore and Paris (1991).

In this example, the applicable expert system goals, listed in order from most to least specific, are:

```
apply-SETQ-to-SETF-transformation
apply-local-transformations-whose-rhs-use-is-more-general-than-lhs-use
apply-local-transformations-that-enhance-maintainability
apply-transformations-that-enhance-maintainability
enhance-maintainability
enhance-program
```

Thus, six possible bindings for the variable ?goal result from the search for domain goals that REPLACE-1 is a step in achieving.

The second constraint of the current plan operator, (GOAL ?hearer ?goal), is a constraint on the user model stating that ?goal must be a goal of the hearer. Not all of the bindings found so far will satisfy this constraint. Those that do not will not be rejected immediately, however, as we do not assume that the user model is complete. Instead, they will be noted as possible bindings, and each will be marked to indicate that, if this binding is used, an assumption is being made, namely that the binding of ?goal is assumed to be a goal of the user. The selection heuristics can be set to tell the planner to prefer choosing bindings that require no assumptions to be made.

In this example, since the user is employing the system to enhance a program and has indicated a desire to enhance the readability and maintainability of the program, the system infers the user shares the top-level goal of the system (ENHANCE-PROGRAM), as well as the two more specific goals ENHANCE-READABILITY and ENHANCE-MAINTAINABILITY. Of these two more specific goals, only ENHANCE-MAINTAINABILITY is on the refinement path leading to the act REPLACE-1. Therefore, the two goals that completely satisfy the first two constraints of the operator shown in Figure 14 are ENHANCE-PROGRAM and ENHANCE-MAINTAINABILITY. Finally, the third constraint indicates that only the most specific goal along any refinement path to the act should be chosen. This constraint encodes the explanation principle that, in order to avoid explaining parts of the reasoning chain that the user is familiar with, when one goal is a subgoal of another, the goal that is lowest in the expert system's refinement structure, i.e., most specific, should be chosen. Note that ENHANCE-MAINTAINABILITY is a refinement of ENHANCE-PROGRAM. Therefore, ENHANCE-MAINTAINABILITY is now the preferred candidate binding for the variable ?goal.

The last two constraints of the operator are also satisfied: REPLACE-1 is the current focus, and the operator is being used to expand a satellite branch of the text plan.

The plan operator is thus instantiated with ENHANCE-MAINTAINABILITY as the binding for the variable ?goal. The selected plan operator is recorded as such, and all other candidate operators are recorded as untried alternatives.

The nucleus of the chosen plan operator is now posted, resulting in the subgoal (MOTIVATION REPLACE-1 ENHANCE-MAINTAINABILITY). The plan operator chosen for achieving this goal is the one shown in Figure 16. This operator motivates the replacement by describing differences between the object being replaced and the object replacing it with respect to the user's goal, i.e., by posting the subgoal (BEL USER (SOMEREF (DIFFERENCES-WRT-GOAL SETF-FUNCTION SETQ-FUNCTION ENHANCE-MAINTAINABILITY))). Although there are many differences between SETQ and SETF, only the differences relevant to the domain goal at hand (ENHANCE-MAINTAINABILITY) should be expressed.

The relevant differences are determined in the following way. From the expert system's problem-solving knowledge, the planner determines what roles SETQ and SETF play in achieving the goal ENHANCE-MAINTAINABILITY. In this case, the system is enhancing maintainability by applying transformations that replace a specific construct with one that has a more general usage. SETQ has a more specific usage than SETF, and therefore the comparison between SETQ and SETF should be based on the generality of their usage. Thus, the goal:

```
(BEL USER (SOMEREF (DIFFERENCES-WRT-GOAL SETQ-FUNCTION
                                         SETF-FUNCTION
                                         ENHANCE-MAINTAINABILITY)))
```

posts the single subgoal

```
(BEL USER (REF (DIFFERENCE SETF-FUNCTION SETQ-FUNCTION) USE)).
```

To satisfy this goal, the system uses an operator that informs the user that SETQ can be used to assign a value to a simple variable, and contrasts this with the use of SETF. Focus information is again updated at this point. SETF becomes the current focus, and USE, ASSIGN-TO-GV and its arguments, VALUE and GENERALIZED-VARIABLE, become potential foci.

Finally, the text planner expands the speech act

```
(INFORM SYSTEM USER (USE SETF-FUNCTION ASSIGN-TO-GV))
```

in order to form a specification for the surface generator. In doing so, the system must express the complex concept ASSIGN-TO-GV where

```
ASSIGN-TO-GV = (ASSIGN (OBJECT VALUE)
                       (DESTINATION GENERALIZED-VARIABLE)).
```

When expressing processes such as ASSIGN, the system expresses the process itself, as well as the participants (e.g., OBJECT) involved in and circumstances (e.g., DESTINATION) surrounding the process. In this case, to express the concept ASSIGN-TO-GV, the system will express the assignment action, the object being assigned (i.e., VALUE), and the destination of the assignment (i.e., GENERALIZED-VARIABLE). To understand the final utterance that will be generated, the listener must know the concepts ASSIGN, VALUE, and GENERALIZED-VARIABLE.

When building specifications for complex processes such as ASSIGN-TO-GV, the planner checks each of the fillers (e.g., VALUE, GENERALIZED-VARIABLE) of the roles (e.g., OBJECT, DESTINATION) of the concept (e.g., ASSIGN) to determine if the user knows that

filler. If so, the planner can simply mention that filler concept by name in the generated text. If, on the other hand, the user model does not indicate that the user is familiar with the concept to be mentioned, the planner must either make an assumption that the user knows the concept (if the planner is in terse mode) or post a subgoal to make the hearer know this concept to the front of its current agenda (if the planner is in verbose mode).

In the current example, recall that the planner is in verbose mode and further suppose that the user model indicates that the user knows the following concepts:

```
CAR-FUNCTION
CDR-FUNCTION
SETQ-FUNCTION
CAR-OF-CONS
CDR-OF-CONS
SIMPLE-VARIABLE
ASSIGN
VALUE
```

Thus, the user model indicates that the user knows the concepts ASSIGN and VALUE but has no indication that the user knows the concept GENERALIZED-VARIABLE. As a result, the system posts a subgoal to make the user know this concept, i.e., (KNOW-ABOUT USER (CONCEPT GENERALIZED-VARIABLE)). Since GENERALIZED-VARIABLE is a member of the potential focus list, no special care need be taken to introduce it. This goal can thus be achieved by elaborating on the previous text to define this new term. This is done with a plan operator that describes concepts by stating their class membership and describing their attributes.

The text plan for response 3 of the sample dialogue is now completed, and it is shown in its entirety in Figure 19. Contrast this text plan with the instantiated schema representation for the same utterance shown in Figure 4. Note that in addition to representing rhetorical relations between portions of the text (e.g., MOTIVATION, CONTRAST, and ELABORATION) that are analogous to the rhetorical predicates contained in the schema, the text plan includes the intentional structure of the text (as shown in Figure 5). Although in this case the text span boundaries coincide with the text segment (as defined by Grosz and Sidner) boundaries, this need not always be the case. In RST, a text span is either a minimal unit or a schema application made up of one or more relations between minimal units. The minimal unit is "essentially a clause, except that clausal subjects and complements and restrictive relative clauses are considered parts of their host clause rather than as separate units" (Mann and Thompson 1988, p. 248). Thus, at least at the lowest level of analysis, text spans can be determined on the basis of syntactic structure alone. For Grosz and Sidner, intentions are the basic determiner of discourse segmentation. A segment must have an identifiable **discourse segment purpose (DSP)**, and embedding relationships between segments are a surface reflection of relationships among their associated DSPs (Grosz and Sidner 1986, pp. 177–178). Therefore, text spans and text segments are based on quite different criteria.

In a text plan produced by our system, any subtree headed by a communicative goal (i.e., an intention) corresponds to a discourse segment in Grosz and Sidner's theory. A segment may consist of more than one span. In such cases, the spans will be connected by subject matter RST relations. Such a case would arise, for example, if the system needed to inform the hearer of a procedure involving several steps. The subtree for the entire procedure would be associated with an intention, and the spans stating the steps would be related to one another by SEQUENCE relations. To construct

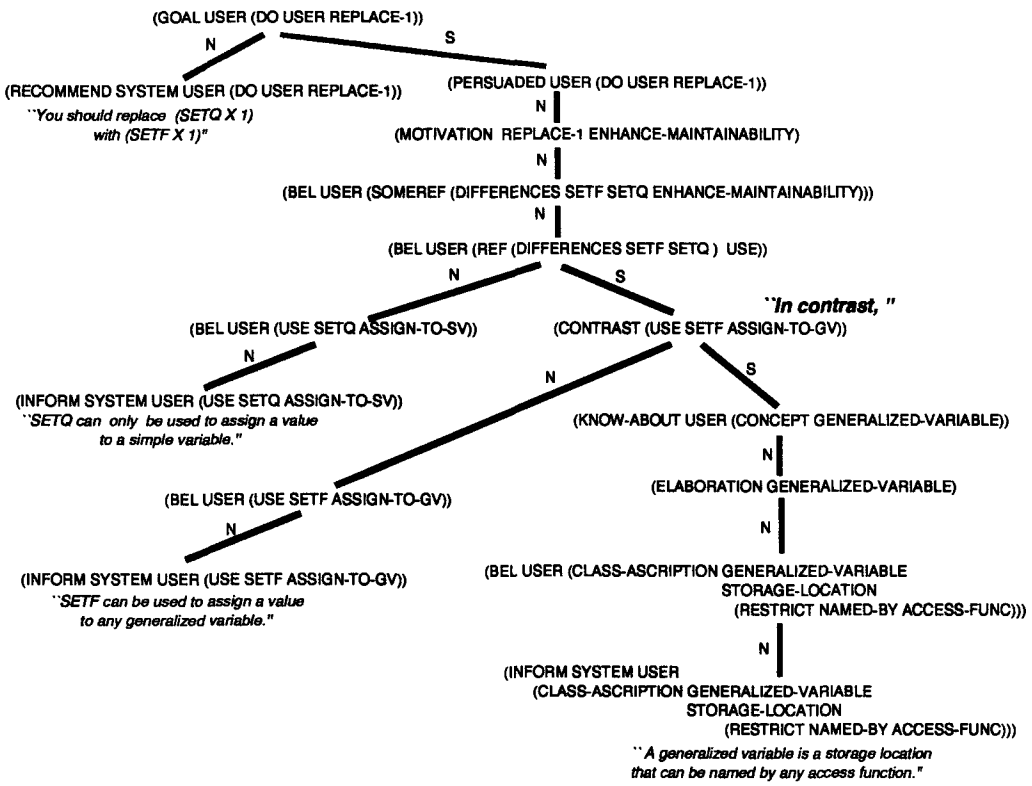

Figure 19
Completed text plan for recommending replace SETQ with SETF.

plan operators for our system based on sample texts, we first segment the texts based on intentional structure. We then identify the rhetorical structure and relate it to the intentional structure. In our formalism, rhetorical techniques are viewed as linguistic strategies for achieving communicative intentions.

In our system, after a text plan like the one shown in Figure 19 is constructed, it is recorded in the system's dialogue history and passed to the grammar interface, which translates the hierarchical text plan into a sequence of sentence specifications suitable for the sentence generator. In the process of this translation, the system decides where to place sentence boundaries and which, if any, connective markers to include. We currently use a simple set of heuristics that make these decisions based on the rhetorical relation between two text spans. To determine sentence boundaries, we have divided the space of RST relations into those that may appear within a clause complex and those that must start a new sentence. To choose connectives, we have associated a small set of possible connectives with each rhetorical relation. Some connectives are suitable for starting new sentences, while others are suitable for constructing complex sentences. If the system finds more than one suitable connective for expressing an RST relation, the first one is chosen.

Thus, in translating the text plan shown in Figure 19 into response 3, we see that `MOTIVATION`, `CONTRAST`, and `ELABORATION` cause a sentence break. Although not shown

in this example, other relations, e.g., MEANS, cause a complex sentence to be formed. In addition, the CONTRAST relation is expressed explicitly via the connective "In contrast," whereas the MOTIVATION and ELABORATION relations do not cause any connectives to be added. The heuristics we have described here are clearly too simplistic, and we are currently working on more sophisticated techniques for linearizing our text plans. However, note that the information recorded in our text plan is already useful in making these decisions, and we believe this information will also facilitate more complex strategies.

6.1 Recovering From Failure: Avoiding Repetition

After the system has produced its recommendation, suppose that the user asks

[USER] What is a generalized variable? [4]

Recall that, from our analysis of naturally occurring dialogues such as the one shown in Figure 1, we observed that advice-seekers frequently ask such questions.

The query analyzer interprets this question and formulates the communicative goal: (KNOW-ABOUT USER (CONCEPT GENERALIZED-VARIABLE)). At this point, the explainer must recognize that this goal was attempted by the last sentence of the previous explanation and was not fully achieved. Failure to do so might lead to simply repeating the description of a generalized variable that the user did not understand. Note that this is precisely what would occur if we had generated the previous explanation using the schema shown in Figure 4. From the schema, the system would not be able to recognize that part of the text previously generated was intended to make the user know about generalized variables. A schema to define a term would thus be triggered and it would give the same answer as was previously generated. Even if a schema-based system were to keep track of the information it already mentioned so that it could avoid literally repeating the same content, it would still not be capable of generating a new text, *taking into consideration the fact that a goal was previously attempted but failed*. The same argument can be made about a system that generates explanations based solely on "RST plans" of the type used in Hovy's (1991) structurer.

By examining the text plan of the previous explanation recorded in the dialogue history, our system is able to determine whether the current goal (resulting from the follow-up question) is a goal that was previously attempted, as it is in this case. This time, when attempting to achieve the goal, the planner must select an alternative strategy. **Recovery heuristics** are responsible for selecting an alternative strategy when responding to such follow-up questions (Moore and Swartout 1989; Moore in press). One of these indicates that when the goal is to make the user know a concept, a good recovery strategy is to give examples.

In the current case, the user model indicates that there are examples of the concept GENERALIZED-VARIABLE that the user is familiar with, namely CAR-OF-CONS and CDR-OF-CONS. Thus, the strategy of giving examples can be applied to yield the following system response:

[SYSTEM] For example, the car of a cons is a generalized variable named by the access function CAR, and the cdr of a cons is a generalized variable named by the access function CDR. [5]

Providing an alternative explanation would not be possible without the explicit representation of the intentional structure that underlies the generated text recorded in our text plans. To avoid repetitions, the system must realize what goals it has tried to achieve in the previous discourse and what strategies it used to achieve them. Then, if

the same goal is posted later in the discourse, the system can realize that its previous strategy was not successful and can employ an alternative strategy.

A similar situation would arise if the user were to indicate that he or she has not been persuaded to replace SETQ with SETF, e.g., with an utterance such as:[13]

[USER] I don't see why I should replace SETQ with SETF. [4']

This would cause the query analyzer to post a goal to persuade the user to perform this act. Once again, the system must recognize that it has already attempted to persuade the user to perform the act by contrasting the usage of SETQ with the usage of SETF. Since this strategy did not succeed, it must be able to persuade the user in a different way. From the information recorded in the text plan in Figure 19, our system can determine that it has already tried to persuade the user to do the replacement. Since MOTIVATION is a presentational RST relation, the system has no other strategies for achieving the persuade goal. However, it does have several other rhetorical strategies for MOTIVATION. One of these is to provide a trace of the expert system's reasoning and could be used to generate the following explanation:

[SYSTEM] I'm trying to enhance the maintainability of the program by applying transformations that enhance maintainability. A transformation enhances maintainability if the usage of the construct on its right hand side is more general than usage of the construct on its left hand side. The usage of SETF is more general than the usage of SETQ. [5']

Again, a schema-based system would not be able to recover correctly from this failure. As we argued earlier, because the schemata do not record the intentions of the schema components, the system cannot determine that it contrasted SETQ with SETF in order to persuade the user to perform the replacement, and thus would not know what part of the schema to replan nor what other strategies to try.

7. Comparison to Related Work

Building on our work, Maybury (1992) devised a system to plan "communicative acts." Like Appelt (1985, p. 9), Maybury's system makes use of a hierarchy of linguistic actions. At the highest level, Maybury has added rhetorical acts (e.g., describe, explain, convince). The next two levels correspond to the top two levels of Appelt's hierarchy: illocutionary acts (e.g., inform, request), and locutionary acts (assert, ask, command).[14] The actions at each level in the hierarchy have been encoded into plan operators that are used in a process of hierarchical decomposition (Sacerdoti 1977) to refine rhetorical acts through illocutionary acts into locutionary acts.

An example operator from Maybury's system is shown in Figure 20. This is a rhetorical operator that can be used to define an entity by giving its "logical definition" (the entity's genus and differentia.) Note that operators in Maybury's language

13 We do not currently allow users to ask questions phrased in such a manner because we do not have a sophisticated natural language understanding component. Instead, we have implemented a direct manipulation interface that allows users to use the mouse to point at the noun phrases or clauses in the text that were not understood or accepted. To approximate the query above, the user could highlight the system's original recommendation to replace SETQ with SETF, and select "Why?" from the menu that appears. As a result of interpreting this "Why?", the system posts the communicative goal (PERSUADED USER (DO USER REPLACE-1)); see Moore and Swartout (1990) for more details.

14 Appelt has two layers below locutionary acts that are not included in Maybury's system: concept activation and utterance acts.

contain both a "header" and an "effects" field. The header field designates the type of act (e.g., define, inform) associated with the operator, whereas the effects field specifies the effect(s) that this act is expected to have on the hearer's mental state. To cause the planner to generate a text, a "discourse controller" posts a goal to cause an effect in the hearer's mental state, such as `KNOW-ABOUT(USER, KC-135)`. This goal is matched against the effects field of operators in the plan library, and one is chosen. Therefore, at the top level, Maybury's system has a record of the intention causing the text to be produced. But now consider what happens during goal refinement. When an operator is instantiated, the clauses in its decomposition field are posted as subgoals. Note from Figure 20 that in Maybury's operator language expressions in the decomposition field are not subgoals to affect the hearer's mental state. Rather, they are linguistic actions such as define, inform, and assert. Unless these actions are locutionary, they become subgoals. To achieve an action subgoal, the planner matches the subgoal against the *header* field of operators in the plan library. When an operator is chosen, the propositions in the effects field are recorded, and the actions in the decomposition become further subgoals.

So, in fact Maybury's system does *not* have a record of the intentional structure behind the text it is producing. There are two problems. First, except at the top level, planning is done by matching against the header field of operators, *not against the effects field*. Because there are multiple effects listed for most of the operators, the system cannot know which one is the *intended* effect! Maybury's system thus cannot distinguish between intended effects and side effects. It is crucial that agents be able to distinguish their intentions from the side effects of their actions in order to recover from plan failures (Davis 1979; Bratman 1987). Therefore, Maybury's system does not have the knowledge necessary for recovering from failures, as our system does.

A second problem with Maybury's text plans is that they do not capture the relationship between intentions, i.e., that some intentions are in the plan because they in turn serve other intentions that appear higher in the plan tree. Once again, this is because Maybury's system simply records all the effects of each action. It is impossible to tell from the sets of effects at each level of the decomposition how effects are related to one another. Grosz and Sidner (1986) argue that such relations between intentions are a crucial part of intentional structure. Contrast Maybury's plans with those produced by our system. Our text plans explicitly represent the intended effects of actions and the relationships between these intentions. While we believe that it is useful to represent additional effects of operators, it is crucial to distinguish intended effects from side effects. Therefore, we argue that while Maybury's approach does indeed represent the effects of all of the "communicative acts" in his plans, it does *not* capture intentional structure and therefore cannot be used to recover from communication failures.

In related work, Cawsey (1993) built a system called EDGE that allows the user to interrupt with clarification questions while a text is being generated. EDGE plans extended tutorial explanations about the structure and input/output behavior of simple electrical circuits. This system is novel because it addresses issues of conversation management, such as turn-taking and topic control. The system separates discourse planning rules from content planning rules for this purpose. EDGE plans an explanation at a high level, following a specified curriculum. This plan is fleshed out as the dialogue progresses, causing sentences to be generated. After each sentence is generated, the system pauses to allow the user to supply feedback by choosing an item from a menu.

The EDGE discourse planner maintains an agenda indicating the topics that will be covered later in the explanation. Based on this agenda, the system can recognize when the user is asking about a topic that will be covered eventually, and can make

NAME	define-by-logical-definition
HEADER	Define(*speaker*, *hearer*, *entity*)
CONSTRAINTS	$\exists c$ Superclass(*entity*, c)
PRECONDITIONS	
ESSENTIAL	$\exists c$ Superclass(*entity*, c)$\wedge$ KNOW-ABOUT(*speaker*, c)
DESIRABLE	$\neg$ KNOW-ABOUT(*hearer*, *entity*)
EFFECTS	$\forall x \in$ superclasses(*entity*) KNOW(*hearer*, Superclass(*entity*, x))$\wedge$ $\forall y \in$ differentia(*entity*) KNOW(*hearer*, Differentia(*entity*, y))
DECOMPOSITION	Inform(*speaker*, *hearer*, Logical-Definition(*entity*))

Figure 20
A plan operator for defining (from Maybury [1992]).

comments such as *We'll be getting to that in a moment.* If this is not the case, or if the user insists, EDGE answers the user's question immediately. Once the interruption has been addressed, EDGE alters its subsequent explanation plan based on what took place during the interruption, and proceeds with its explanation as specified in the overall explanation plan. To resume from interruptions coherently, the discourse planning rules that manage the conversation include markers and meta-comments (e.g., *Anyway, I was talking about...*).

Because of its extended explanation plan and its discourse management rules, EDGE can handle interruptions in ways that are beyond the current capability of our system. However, as Cawsey points out, EDGE is based on a largely syntactic model of dialogue structure, and the system does not explicitly represent *why* different dialogue actions are selected. The effects that dialogue operators are intended to have on the user's knowledge and goals are not represented. EDGE content plans are much like schemas and therefore suffer from the limitations we discussed in Section 3.2.1. If a user fails to understand a text produced by a content plan, the system's only recourse is to try another strategy to achieve the top-level content goal, e.g., "describe how an entity works." Moreover, the content plans of EDGE are domain-specific, and are not based on general rhetorical techniques. Because rhetorical structure is not represented, the system cannot choose connective markers or use other rhetorical devices that make text easier to comprehend. For these reasons, EDGE cannot handle the types of phenomenon our system handles.

It is also important to note that, because we are dealing with expert and advisory applications, our system must be able to manage a dialogue whose structure emerges dynamically as the user asks questions. In advisory interactions, the system presents the user with a recommendation or result and only provides explanations when the user requests them. It is not appropriate for the system to plan extended explanations, testing the user's understanding and elaborating without provocation. Therefore, Cawsey's approach, which relies on the fact that the system has an extended explanation plan to follow, cannot be used directly.

It is clear, however, that our approach and Cawsey's are complementary, and that a complete system would need to incorporate aspects of both. In particular, our system should be augmented to include conversation management operators in order to manage topic shifts, to handle interruptions, and to generate meta-comments about the discourse itself.

8. Status and Future Directions

The text planner presented in this paper is implemented in Common Lisp and can produce the text plans necessary to participate in the sample dialogue described in this paper and several others; see Moore (in press) and Paris (1991a). We currently have over 150 plan operators that can answer the following types of questions:

- Why?
- Why *conclusion*?
- Why are you trying to achieve *goal*?
- Why are you using *method* to achieve *goal*?
- Why are you doing *act*?
- How do you achieve *goal* (in the general case)?
- How did you achieve *goal* (in this case)?
- What is a *concept*?
- What is the difference between *concept1* and *concept2*?
- Huh?

The text planner is being incorporated into several knowledge-based systems and two intelligent tutoring systems currently under development. Two of these systems are intended to be installed and used in the field. This will give us an opportunity to evaluate the techniques proposed here and extend the system as appropriate. It has also been employed in Reithinger's (1991) system for incremental language generation, and serves as the basis of the presentational planner for WIP, a multimedia system that plans text and graphics to achieve communicative goals (Wahlster et al. 1991). Finally, it is the basis for a text planner capable of generating explanatory texts that integrate examples with their surrounding context (Mittal and Paris 1993). This integration would not be possible without our system's explicit representation of the intentions for generating portions of the text and the rhetorical strategies used to achieve them.

We have begun to investigate how the discourse history should be indexed and exploited to control the dialogue and affect subsequent responses in more general ways. As reported here, the dialogue history is used primarily to determine how to interpret and answer follow-up questions (e.g., *Why?*, *How come?*), and to determine how to respond when the user asks a question that has already been answered or indicates that an explanation was not understood (*Huh?*). In Carenini and Moore (1993) and Rosenblum and Moore (1993) we discuss additional ways in which prior explanations can affect the generation of the current utterance.

We currently do not allow the user to return to a previous topic (e.g., once the system has moved on to a new topic, *Let's go back to replacing SETQ with SETF* ...), or to introduce new goals into the dialogue (e.g., *Well, now suppose I wanted to enhance efficiency* ...). In order to allow the user to change topics and introduce new goals at will, the system will need to be able to track the user's shifting goals and attention. Sidner (1985) and Carberry (1987) have proposed approaches for tracking the topic of conversation in task-oriented dialogues. However, their approaches rely on the assumption that the topic of conversation closely follows the structure of the domain task. Litman and

Allen (1987) identified types of subdialogues in task-oriented interactions, including clarifications and corrections, in which topic shift deviates from task structure, and they devised a plan recognition model for handling such subdialogues. Our system currently handles what Litman and Allen call **clarification subdialogues**. We believe that our model could be extended to handle other types of subdialogues, and that the text plans recorded in our dialogue history will aid in more general discourse management tasks than the ones we currently address. To perform these tasks, our system must understand how the previous responses stored in its discourse history relate to one another. That is, we must address issues of how to build a representation of the intentional structure of the dialogue that is emerging across conversational turns (Grosz and Sidner 1986) and to track global focus (Grosz 1977). In addition, we will need communicative strategies for managing the dialogue, e.g., strategies for introducing a topic, strategies for returning to a topic, etc.

9. Conclusions

We have presented an approach to natural language generation that extends previous theories and implementations in order to enable a computational system to play the role of a dialogue participant in an advisory setting. We began by illustrating the types of phenomena that are prevalent in advisory dialogues. We argued that, in order to participate in such dialogues, a system must be capable of reasoning about its own previous utterances. Follow-up questions must be interpreted in the context of the ongoing conversation, and the system's previous contributions form part of this context.

We claimed that to handle explanation dialogues successfully, a discourse model must include the intended effect of individual parts of the text on the hearer, as well as a representation of how the parts relate to one another rhetorically. Through principled arguments and detailed examples, we showed that previous approaches to multisentential text generation, which do not explicitly represent the intentional structure of their utterances, cannot be used for advisory dialogues. We presented our approach to text generation in which the system reasons about and records the intentions behind each text span as well as the rhetorical means used to achieve them. Finally, we demonstrated how this record can be used to overcome some of the limitations of earlier approaches.

Acknowledgments

The research described in this paper was supported in part by the Advanced Research Projects Agency under a NASA Ames cooperative agreement Number NCC 2-520. Johanna Moore is currently supported by grants from the Office of Naval Research Cognitive and Neural Sciences Division (Grant Number N00014-91-J-1694), the National Science Foundation Research Initiation Award (Grant Number IRI-9113041), and the National Library of Medicine. Cécile Paris gratefully acknowledges the support of the Advanced Research Projects Agency under the contract DABT63-91-C-0025 and the National Science Foundation (Grant Number IRI-9003078) while writing this paper.

The authors would like to thank William Swartout, who has advised us on this work since its inception and has also given us useful comments on this paper. We would also like to thank Martha Pollack for her help in clarifying some of the ideas presented in this paper, and Giuseppe Carenini, Violetta Cavalli-Sforza, Vibhu Mittal, Richmond Thomason, and Arlene Weiner, who provided useful comments on earlier drafts of this paper. Finally, we are indebted to the anonymous reviewers whose detailed and insightful comments greatly improved the final version of this paper.

References

Appelt, Douglas E. (1985). *Planning English Sentences*. Cambridge University Press.

Bateman, John A., and Paris, Cécile L. (1989). "Phrasing a text in terms the user can understand." In *Proceedings, Eleventh International Joint Conference on Artificial Intelligence*, 1511–1517. Detroit, MI.

Bateman, John A., and Paris, Cécile L. (1991). "Constraining the deployment of lexicogrammatical resources during text generation: Towards a computational instantiation of register theory." In *Functional and Systemic Linguistics: Approaches and Uses*, edited by Eija Ventola, 81–106. Mouton de Gruyter.

Bratman, Michael (1987). *Intentions, Plans, and Practical Reason*. Harvard University Press.

Brewer, W. F. (1980). "Literacy theory, rhetoric, and stylistics: Implications for psychology." In *Theoretical Issues in Reading Comprehension*, edited by R. J. Shapiro, B. C. Bruce, and W. F. Brewer, 221–239. Lawrence Erlbaum.

Cahour, Béatrice; Falzon, Pierre; and Robert, Jean Marc (1990). "From text coherence to interface consistency: A psycholinguistic approach." In *Work with Display Units 89*, edited by L. Berlinguet and D. Berthelette. Elsevier Science Publishers B.V.

Carberry, Sandra M. (1987). "Pragmatic modeling: Toward a robust natural language interface." *Computational Intelligence*, 3(3), 117–136.

Carenini, Giuseppe, and Moore, Johanna D. (1993). "Generating explanations in context." In *Proceedings, International Workshop on Intelligent User Interfaces*, edited by Wayne D. Gray, William E. Hefley, and Dianne Murray, 175–182. ACM Press.

Cawsey, Alison (1993). *Explanation and Interaction: The Computer Generation of Explanatory Dialogues*. MIT Press.

Charney, Davida H.; Reder, Lynne M.; and Wells, Gail W. (1988). "Studies of elaboration in instructional texts." In *Effective Documentation: What We Have Learned from Research*, edited by Stephen Doheny-Farina, 48–72. MIT Press.

Clancey, William J. (1983). "The epistemology of a rule-based expert system: A framework for explanation." *Artificial Intelligence*, 20(3), 215–251.

Cohen, Philip R. (1978). *On knowing what to say: Planning speech acts*. Doctoral dissertation, Department of Computer Science, University of Toronto.

Cohen, Philip R., and Levesque, Hector (1990). "Rational interaction as the basis for communication." In *Intentions in Communication*, edited by Philip R. Cohen, Jerry Morgan, and Martha E. Pollack, 221–255. MIT Press

Cohen, Philip R., and Perrault, C. Raymond (1979). "Elements of a plan-based theory of speech acts." *Cognitive Science*, 3, 177–212.

Davis, Lawrence H. (1979). *Theory of Action*. Prentice Hall.

Ehrlich, Marie-France, and Cahour, Béatrice (1991). "Contrôle métacognitif de la compréhension: cohésion d'un texte expositif et auto-évaluation de la compréhension." *Bulletin de Psychologie*, XLIV(399), 147–155. Special edition edited by E. Cauzinille and J. Beaudichon.

Goldman, Susan, and Durán, Richard P. (1988). "Answering questions from oceanography texts: Learner, task, and text characteristics." *Discourse Processes*, 1, 373–412.

Grimes, Joseph E. (1975). *The Thread of Discourse*. Mouton.

Grosz, Barbara J. (1977). "The representation and use of focus in dialogue understanding." Technical Report 151, SRI International, Menlo Park, CA.

Grosz, Barbara J., and Sidner, Candace L. (1986). "Attention, intention, and the structure of discourse." *Computational Linguistics*, 12(3), 175–204.

Hobbs, Jerry R. (1979). "Coherence and coreference." *Cognitive Science*, 3(1), 67–90.

Hobbs, Jerry R. (1983). "Why is discourse coherent?" In *Coherence in Natural Language Texts*, edited by F. Neubauer, 29–69. H. Buske.

Hobbs, Jerry R. (1985). "On the coherence and structure of discourse." Technical Report CSLI-85-37, Center for the Study of Language and Information, Leland Stanford Junior University, Stanford, California.

Hovy, Eduard H. (1988). *Generating Natural Language Under Pragmatic Constraints*. Lawrence Erlbaum.

Hovy, Eduard H. (1991). "Approaches to the planning of coherent text." In *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, edited by Cécile L. Paris, William R. Swartout, and William C. Mann, 83–102. Kluwer Academic Publishers.

Kittredge, R.; Korelsky, T.; and Rambow, O. (1991). "On the need for domain communication knowledge." *Computational Intelligence*, 7(4), 305–314.

Klausmeier, Herbert J. (1976). "Instructional design and the teaching of concepts." In *Cognitive Learning in Children*, edited by J. R. Levin and V. L. Allen. Academic Press.

Levy, David M. (1979). "Communicative

goals and strategies: Between discourse and syntax." In *Syntax and Semantics, Volume 12: Discourse and Syntax*, edited by P. Cole and J. L. Morgan, 183–210. Academic Press.

Litman, Diane J., and Allen, James F. (1987). "A plan recognition model for subdialogues in conversations." *Cognitive Science*, 11, 163–200.

Mann, William C. (1984). "Discourse structures for text generation." In *Proceedings, Tenth International Conference on Computational Linguistics*, 367–375. Stanford, CA.

Mann, William C., and Matthiessen, Christian M. I. M. (1985). "A demonstration of the Nigel text generation computer program." In *Systemic Perspectives on Discourse: Selected Papers from the Ninth International Systemics Workshop*, edited by R. Benson and J. Greaves, 50–83. Ablex.

Mann, William C., and Thompson, Sandra A. (1988). "Rhetorical structure theory: Towards a functional theory of text organization." *TEXT*, 8(3), 243–281.

Maybury, Mark T. (1992). "Communicative acts for explanation generation." *International Journal of Man-Machine Studies*, 37(2), 135–172.

McCoy, Kathleen F. (1989). "Generating context sensitive responses to object-related misconceptions." *Artificial Intelligence*, 41(2), 157–195.

McKeown, Kathleen R. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press.

Meyer, B. J. F.; Brandt, D. M.; and Bluth, G. J. (1980). "Use of top-level structure in texts: Key for reading comprehension in ninth-grade students." *Reading Research Quarterly*, 16, 72–102.

Mittal, Vibhu O., and Paris, Cécile L. (1993). "Automatic documentation generation: The interaction between text and examples." In *Proceedings, Thirteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Chambery, France.

Moore, Johanna D. (In press.) *Participating in Explanatory Dialogues: Interpreting and Responding to Questions in Context*. MIT Press.

Moore, Johanna D., and Paris, Cécile L. (1991). "Requirements for an expert system explanation facility." *Computational Intelligence*, 7(4), 367–370.

Moore, Johanna D., and Paris, Cécile L. (1992). "Exploiting user feedback to compensate for the unreliability of user models." *User Modeling and User-Adapted Interaction*, 2(4), 331–365.

Moore, Johanna D., and Pollack, Martha E. (1992). "A problem for RST: The need for multi-level discourse analysis." *Computational Linguistics*, 18(4), 537–544.

Moore, Johanna D., and Swartout, William R. (1989). "A reactive approach to explanation." In *Proceedings, Eleventh International Joint Conference on Artificial Intelligence*, 1504–1510. Detroit, MI.

Moore, Johanna D., and Swartout, William R. (1990). "Pointing: A way toward explanation dialogue." In *Proceedings, National Conference on Artificial Intelligence*, 457–464. Boston, MA.

Neches, Robert; Swartout, William R.; and Moore, Johanna D. (1985). "Enhanced maintenance and explanation of expert systems through explicit models of their development." *IEEE Transactions on Software Engineering*, SE-11(11), 1337–1351.

Paris, Cécile L. (1988). "Tailoring object descriptions to the user's level of expertise." *Computational Linguistics*, 14(3), 64–78.

Paris, Cécile L. (1991a). "Generation and explanation: Building an explanation facility for the explainable expert systems framework." In *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, edited by Cécile L. Paris, William R. Swartout, and William C. Mann, 49–81. Kluwer Academic Publishers.

Paris, Cécile L. (1991b). *The Use of Explicit User Models in Text Generation: Tailoring to a User's Level of Expertise*. Frances Pinter.

Penman Natural Language Generation Group (1989). The Penman User Guide. Available from USC/Information Sciences Institute, 4676 Admiralty Way, Marina del Rey, CA.

Rambow, Owen (1990). "Domain communication knowledge." In *Proceedings, Fifth International Workshop on Natural Language Generation*, 87–94. Pittsburgh, PA

Redeker, Gisela (1990). "Ideational and pragmatic markers of discourse structure." *Journal of Pragmatics*, 14, 367–381.

Reithinger, Norbert (1991). *Eine parallele Architektur zur inkrementellen Generierung multimodaler Dialogbeiträge*. Doctoral dissertation, Technischen Fakultät der Universität des Saarlandes, Saarbrücken, Germany.

Robinson, Jane J. (1984). "Extending grammars to new domains." Technical Report ISI/RR-83-123, USC/Information

Sciences Institute.
Rosenblum, James A., and Moore, Johanna D. (1993). "Participating in instructional dialogues: Finding and exploiting relevant prior explanations." In *Proceedings, World Conference on Artificial Intelligence in Education.*
Sacerdoti, Earl D. (1977). *A Structure for Plans and Behavior.* Elsevier.
Schiffrin, Deborah (1987). *Discourse Markers.* Cambridge University Press.
Shepherd, H. R. (1926). *The Fine Art of Writing.* Macmillan.
Sidner, Candace L. (1979). *Toward a computational theory of definite anaphora comprehension in English discourse.* Doctoral dissertation, Massachusetts Institute of Technology, Cambridge, MA.
Sidner, Candace L. (1985). "Plan parsing for intended response recognition in discourse." *Computational Intelligence,* 1(1), 1–10.
Sparck Jones, Karen (1989). "Realism about user modelling." In *User Models in Dialog Systems,* edited by Alfred Kobsa and Wolfgang Wahlster, 341–363. Symbolic Computation Series, Springer-Verlag.
Suthers, Daniel D. (1991). "Task-appropriate hybrid architectures for explanation." *Computational Intelligence,* 7(4), 315–333.
Swartout, William R. (1983). "XPLAIN: A system for creating and explaining expert consulting systems." *Artificial Intelligence,* 21(3), 285–325.
Swartout, William R.; Paris, Cécile L.; and Moore, Johanna D. (1991). "Design for explainable expert systems." *IEEE Expert,* 6(3), 58–64.
Wahlster, Wolfgang; André, Elisabeth; Graf, Winfried; and Rist, Thomas (1991). "Designing illustrated texts: How language production is influenced by graphics generation." In *Proceedings, European Chapter of the Association for Computational Linguistics,* 8–14. Berlin.